

LUCIANO SILVA

SEGMENTAÇÃO DE IMAGENS DE PROFUNDIDADE POR DETECÇÃO DE BORDAS

*Dissertação apresentada como requisito parcial
à obtenção do grau de Mestre. Programa de
Pós-Graduação em Informática. Setor de Ciências
Exatas, Universidade Federal do Paraná.*

Orientadora: Prof.^a Dr.^a Olga R. P. Bellon

CURITIBA

2000



Ministério da Educação
Universidade Federal do Paraná
Mestrado em Informática

PARECER

Nós, abaixo assinados, membros da Banca Examinadora da defesa de Dissertação de Mestrado em Informática do aluno **Luciano Silva**, avaliamos o trabalho intitulado "**Segmentação de Imagens de Profundidade por Detecção de Bordas**", cuja defesa foi realizada no dia 04 de outubro de 2000. Após a avaliação, decidimos pela aprovação do Candidato.

Curitiba, 04 de outubro de 2000.

Prof.^a Dra. Olga Regina Pereira Bellon
Presidente - DINF/UFPR

Prof. Dr. Luciano da Fontoura Costa
IFSC/USP

Prof. Dr. Alexandre Ibrahim Direne
DINF/UFPR

Prof. Dr. Hélio Pedrini
DINF/UFPR

LUCIANO SILVA

**SEGMENTAÇÃO DE IMAGENS DE PROFUNDIDADE POR
DETEÇÃO DE BORDAS**

Dissertação aprovada como requisito parcial à obtenção do grau de
Mestre no Programa de Pós-Graduação em Informática da Universidade
Federal do Paraná, pela Comissão formada pelos professores:

Orientadora: Profa. Dra. Olga R. P. Bellon
Departamento de Informática, UFPR

Prof. Dr. Luciano Fontoura
Instituto de Física, USP

Prof. Dr. Hélio Pedrini
Departamento de Informática, UFPR

Prof. Dr. Alexandre I. Direne
Departamento de Informática, UFPR

Curitiba, 04 de outubro de 2000

AGRADECIMENTOS

Agradeço à todos que colaboraram para que eu conquistasse este objetivo, subindo mais um degrau para a consolidação da minha carreira de pesquisador.

Agradeço à participação dos professores, em especial do Prof. Dr. Alexandre Direne pela ajuda contribuindo para o enriquecimento dos meus conhecimentos na vida acadêmica, e do professor Hélio Pedrini pelos perfeitos comentários e pontuais ajustes para melhorar a versão final desta dissertação de mestrado.

À minha e só minha Olguinha, pela orientação, paciência e compreensão, companheira tanto nos momentos agradáveis quanto nos mais difíceis, um grande beijo e, muito obrigado pelas incontáveis contribuições na minha formação acadêmica e profissional.

À meus pais, Luir e Maria Sílvia, pelo apoio inexcedível e imensurável ajuda em todos estes anos da minha vida e principalmente neste últimos anos de estudo, meu sincero muito obrigado.

Muito obrigado meus irmãos, Fabiano e Fernando pelo apoio. Desejo toda a sorte e felicidade deste mundo à vocês. Agradeço também a meus familiares e amigos por estarem sempre presentes.

E, Vó Nair, esteja onde estiver, sei que você está feliz assim como eu estou agora, um abraço bem apertado e um grande beijo.

Luciano Silva

SUMÁRIO

LISTA DE FIGURAS	vi
RESUMO	vii
ABSTRACT	viii
1 INTRODUÇÃO	1
2 CONCEITOS BÁSICOS	8
2.1 Visão humana e visão computacional	8
2.2 Imagens de intensidade luminosa	9
2.3 Imagens de profundidade	11
2.3.1 ABW	12
2.3.2 Perceptron	15
2.4 Imagens binárias	17
2.5 Máscaras	17
2.5.1 Vizinhaça	18
2.5.2 Filtros	19
2.6 Morfologia Matemática	20
2.7 Gradientes	22
2.8 Bordas	23
2.9 Limiares	24
2.10 Segmentação de imagens	26
2.10.1 Detecção de bordas	26
2.10.1.1 Canny	28
2.10.1.2 Rothwell	29
2.10.1.3 Nalwa	30
2.10.2 Métodos de agrupamento	30

2.10.3	Crescimento de regiões	32
2.10.4	Divisão-e-união	33
2.11	Afinamento	33
2.12	Fechamento	35
3	SEGMENTAÇÃO DE IMAGENS DE PROFUNDIDADE	37
3.1	Definição do problema	37
3.2	Trabalhos relacionados	38
3.2.1	USF	38
3.2.2	WSU	40
3.2.3	UB	42
3.2.4	UE	44
3.3	Métodos de avaliação	46
3.4	Conclusão	51
4	SEGMENTAÇÃO DE IMAGENS DE PROFUNDIDADE POR DETEÇÃO DE BORDAS	53
4.1	Objetivos	53
4.2	Método desenvolvido	54
4.2.1	Filtragem	54
4.2.2	Deteção de bordas	55
4.2.2.1	Bordas de profundidade	55
4.2.2.2	Bordas de orientação	58
4.2.3	Refinamento do mapa de bordas	63
4.2.4	Crescimento de regiões	69
4.3	Avaliação	70
4.4	Conclusão	71
5	RESULTADOS EXPERIMENTAIS	73
5.1	Avaliação dos algoritmos da literatura	73
5.2	Avaliação do método desenvolvido	75

	iv
5.3 Resultados comparativos	81
5.4 Conclusão	94
6 CONCLUSÃO	95
BIBLIOGRAFIA	102

LISTA DE FIGURAS

2.1	Objeto iluminado por feixes de luz com diferentes padrões.	13
2.2	Processo de aquisição da informação de profundidade dos objetos.	13
2.3	Imagem do tipo ABW.	15
2.4	Imagem do tipo Perceptron.	16
2.5	Tipos de vizinhança mais utilizados (vizinhança 4 e 8).	18
2.6	Processo de filtragem pelo filtro média e filtro mediano em uma imagem de profundidade ABW.	20
2.7	Exemplo de 3 elementos estruturantes.	21
2.8	Exemplo das 4 variações de um elemento estruturante (0, 90, 180 e 270 graus).	21
2.9	Diferentes tipos de bordas de uma imagem de profundidade.	23
2.10	Exemplo de segmentação de imagens de profundidade.	31
4.1	Resultado do processo de crescimento de regiões dos <i>pixels</i> válidos.	59
4.2	Situação onde deve ser tomado cuidado na marcação dos <i>pixels</i> válidos para o cálculo do vetor normal de $p(x, y)$	60
4.3	<i>Pixels</i> válidos marcados no processo de crescimento de regiões para o cálculo do vetor normal.	60
4.4	Crescimento de regiões dos <i>pixels</i> válidos para o cálculo do vetor normal em $p(x, y)$, para situações com <i>pixels</i> de borda de extremidade.	61
4.5	<i>Pixels</i> considerados ruídos nas bordas. Parasitas, representados por k e os buracos representados por q	64
4.6	Elementos estruturantes para a eliminação de ruídos nas bordas.	64
4.7	Processo de esqueletização do mapa de bordas	65
4.8	Diferença da erosão por vizinhança 4 e por vizinhança 8 no processo de esqueletização.	66
4.9	Elementos estruturantes utilizados no processo de erosão.	66

4.10 Região de bordas do resultado do processo de esqueletização com *pixels* redundantes. 67

4.11 Elementos estruturantes utilizados no processo de eliminação de *pixels* redundantes. 67

4.12 Processo de fechamento do mapa de bordas de uma região com um *pixel* de extremidade. 68

4.13 Processo de fechamento do mapa de bordas de uma região com vários *pixels* de extremidade. 69

5.1 Etapas do processo de segmentação de imagens de profundidade por detecção de bordas em uma imagem de profundidade do tipo ABW. 78

5.2 Exemplo de segmentação de uma imagem de profundidade com um objeto não poliédrico baseado na detecção de bordas. 79

5.3 Imagem de treinamento ABW (abw.train.0). 82

5.4 Imagem de treinamento ABW (abw.train.1). 83

5.5 Imagem de treinamento ABW (abw.train.2). 84

5.6 Imagem de treinamento ABW (abw.train.3). 85

5.7 Imagem de treinamento ABW (abw.train.4). 86

5.8 Imagem de treinamento ABW (abw.train.5). 87

5.9 Imagem de treinamento ABW (abw.test.1). 88

5.10 Imagem de treinamento ABW (abw.test.3). 89

5.11 Imagem de treinamento ABW (abw.test.4). 90

5.12 Imagem de treinamento ABW (abw.test.8). 91

5.13 Imagem de treinamento ABW (abw.test.10). 92

5.14 Imagem de treinamento ABW (abw.test.18). 93

RESUMO

Esse trabalho é uma contribuição para a solução do problema da segmentação de imagens de profundidade. Para isso, foi desenvolvido um método de segmentação baseado em técnicas de detecção de bordas. Os objetivos principais do método desenvolvido são: preservar a topologia dos objetos observados na cena, ter bom desempenho na presença de ruídos e reduzir ao mínimo a dependência dos algoritmos utilizados em relação a valores rígidos de limiar. Esses objetivos foram direcionados de forma que a segmentação resultante possa ser adequadamente utilizada para processos de representação de objetos. Com isso, espera-se também contribuir para o desenvolvimento de sistemas que permitam a interpretação automática de objetos. Os resultados apresentados foram comparados com os resultados de outros 4 segmentadores avaliados que comprovaram algumas contribuições da metodologia desenvolvida baseada em detecção de bordas. Os principais pontos positivos do método desenvolvido podem ser observados nos resultados da segmentação de imagens de profundidade, que mostraram uma segmentação mais precisa, preservando a topologia das superfícies dos objetos. Nestes resultados a eliminação dos ruídos nas imagens segmentadas também pode ser considerada um ponto positivo neste método. Além disso, a eliminação de muitos limiares fixos ajuda na generalização do método para ser aplicado a diversos tipos de imagens de entrada. Finalmente, ressalta-se que o método não é restrito à aplicação em imagens compostas apenas por objetos polédricos.

ABSTRACT

This work contributes to solve the range image segmentation problem where the solution method is based on edge detection techniques. The main objectives are: to preserve the object topology, to be performant even for noisy images, and to minimize the algorithm's dependency on rigid threshold values. These objectives were chosen in order to provide suitable segmentation results to build object representations, which can enhance the image system task of automatic object interpretation. The presented results were compared to other four segmentation algorithms, and corroborate some contributions of the developed methodology. The main advantages of the range image segmentation by edge detection techniques can be observed in the segmentation results, which show that the object's topology were preserved and the influence of image noise was eliminated in a wide range of situations. Another advantage is that the developed method has also eliminated many traditional rigid threshold values, which helps its generalization and use in many different image types. Finally, it must be emphasized that the method is not limited for application only on images of polihedral objects.

CAPÍTULO 1

INTRODUÇÃO

A maioria dos sistemas de visão computacional desenvolvidos nos últimos anos utilizam imagens de intensidade luminosa (*intensity images*). Os sensores para captação destas imagens têm baixo custo e geram dados de boa qualidade. Entretanto, existe uma grande limitação neste tipo de imagens, que é a ausência de informação tridimensional (3D) sobre a cena observada, o que dificulta a sua interpretação.

Imagens de profundidade (*range images*) fornecem a informação 3D diretamente [6], favorecendo a extração de características e sua posterior interpretação. Isto justifica a atenção que essas imagens vêm recebendo nos últimos anos. Porém, apesar desta vantagem, vários problemas inerentes ao processo de análise de imagens continuam como de difícil solução. Isso ocorre, principalmente, na etapa de segmentação, que é uma tarefa essencial para a interpretação de imagens e depende diretamente da correta extração de suas características.

A etapa de segmentação tem como principal atribuição identificar e isolar partes homogêneas significativas da imagem. Para isso, a maior limitação é a necessidade de se definir valores de limiar rígidos para a tomada de decisões. Tipicamente, utiliza-se dados empíricos ou heurísticos para contornar esta dificuldade. Por este motivo, para se obter resultados experimentais expressivos, até o presente momento, é necessário um grande domínio sobre o processo de obtenção das imagens e sobre o modelo dos objetos utilizados nos experimentos [21].

Duas principais abordagens podem ser aplicadas ao problema da segmentação: segmentação por regiões e segmentação por detecção de bordas. No primeiro caso, existe a busca por conjuntos de elementos da imagem (*pixels*) com características homogêneas, de acordo com o predicado de homogeneidade definido que, reunidos, devem formar uma região. Para a segmentação por bordas, a proposta é complementar e a busca é por *pi-*

xels da imagem que apresentem descontinuidade significativa de valores em relação a seus vizinhos, indicando a presença de um limite entre duas regiões.

A segmentação por regiões é realizada, principalmente, com o uso das seguintes técnicas [10, 12, 22]: agrupamento (*clustering*), crescimento de regiões (*region growing*) e divisão-e-união (*split-and-merge*). O resultado de um processo de segmentação por regiões é uma imagem rotulada, ou seja, cada *pixel* possui um valor que o associa a uma das regiões definidas pelo processo.

A segmentação por detecção de bordas, que normalmente oferece como resultado uma imagem binária, sempre foi um tema que recebeu muita atenção dos pesquisadores. Porém, apesar de todo o esforço aplicado a este problema, ele está longe de ter uma solução fechada. Ao contrário, em relação às imagens de profundidade, que abriram uma série de novos desafios a serem ultrapassados, as soluções estão apenas começando a despontar.

Com esse objetivo, em relação à segmentação por detecção de bordas, um grande esforço vem sendo aplicado no desenvolvimento de técnicas para preservar as junções dos objetos observados na imagem. Tais junções são características geométricas importantes que, se forem corretamente identificadas garantem, por exemplo, informação relevante sobre a topologia dos referidos objetos [5, 19]. Aliado a isso, existe a busca por soluções que sejam imunes a ruídos, que é um grande problema ainda encontrado nas imagens geradas pela maioria dos sensores de profundidade utilizados em instituições de pesquisa. Os resultados desta linha de pesquisa têm implicações diretas na confiabilidade dos modelos obtidos para os objetos, o que pode melhorar a interpretação das imagens.

Considerando-se as duas opções possíveis para segmentação, por bordas e por regiões, é sabido que ambas resultam em falhas na identificação das regiões, quando aplicadas a sistemas de visão computacional. Tendo em vista que as abordagens são complementares, intuitivamente observa-se que a combinação de seus resultados pode levar a uma solução mais robusta para o problema. De fato, essa combinação já vem sendo explorada por alguns pesquisadores [4, 5, 8, 10].

Esse trabalho é uma contribuição para a solução do problema da segmentação de

imagens de profundidade. Para isso, foi desenvolvido um método de segmentação baseado em técnicas de detecção de bordas. Os objetivos principais do método desenvolvido são: preservar a topologia dos objetos observados na cena, ter bom desempenho na presença de ruídos e reduzir ao mínimo a dependência dos algoritmos utilizados em relação a valores rígidos de limiar. Esses objetivos foram direcionados de forma que a segmentação resultante possa ser adequadamente utilizada para processos de representação de objetos. Com isso, espera-se também contribuir para o desenvolvimento de sistemas que permitam a interpretação automática de objetos.

Dentro do contexto apresentado, a primeira fase do trabalho foi um estudo para formalização do problema da detecção de bordas em imagens. Foram estudados artigos clássicos sobre o assunto, tais como os de Marr e Hildreth [29], Canny [9], Deriche [11], Nalwa [32], Mintz [30] e Health [19], entre outros. Embora esses trabalhos sejam voltados a imagens de intensidade luminosa, eles formaram a base para o correto entendimento do problema.

Na sequência, o estudo fixou-se mais em técnicas para detecção de bordas em imagens de profundidade. Foram levantados diversos trabalhos sobre o assunto: Mitiche e Aggarwal [31], Krishnapuram e Gupta [26], Jiang e Bunke [24] e Mintz [30], porém nenhum deles apresentou ainda uma solução adequada para o problema. As principais deficiências encontradas são: grande sensibilidade a ruídos, pouca precisão na localização da borda e perda de informação sobre a topologia dos objetos.

A grande maioria dos trabalhos para detecção de bordas em imagens de profundidade identificam dois principais tipos de bordas: bordas de profundidade e bordas de orientação. As bordas de profundidade são as regiões da imagem onde existe uma descontinuidade significativa de profundidade. Já nas bordas de orientação, a descontinuidade se refere à orientação das superfícies. Dada a natureza diferente das duas informações, é necessário aplicar técnicas independentes para detectar cada tipo de borda.

A identificação das bordas de profundidade apresenta uma problemática semelhante à da detecção de bordas em imagens de intensidade luminosa. Para identificar bordas de orientação, entretanto, existe uma dificuldade a mais associada que é a necessidade de se determinar a orientação das superfícies para cada *pixel* da imagem. Na verdade, esse é o

maior limitador para a obtenção de soluções precisas, como é discutido no decorrer deste trabalho.

Afastando-se um pouco da limitação do escopo em métodos para detecção de bordas, e abrindo o enfoque em direção à segmentação de imagens de profundidade de uma forma geral, deparou-se com alguns bem organizados grupos de pesquisa sobre o assunto, cujos trabalhos foram devidamente estudados. Esses grupos, entre outras contribuições, desenvolveram métricas para comparação de seus segmentadores, publicadas em Hoover et al. [21], bem como disponibilizam um *site*¹ na *internet* com informações específicas sobre o tema. Em decorrência deste trabalho, concluiu-se que nem mesmo a segmentação de objetos compostos apenas por superfícies planares está perto de ter uma solução fechada.

O interesse principal do presente trabalho destoa ligeiramente daqueles apresentados por Hoover, o que impossibilitou que uma comparação numérica direta fosse realizada de imediato. Entretanto, a avaliação dos algoritmos apresentados, bem como as imagens e os resultados disponibilizados no referido *site*, serviram como base para guiar a apresentação deste trabalho. No decorrer desse documento, também são apresentadas discussões relevantes sobre esses algoritmos de segmentação e comparações com os resultados obtidos com a aplicação do método desenvolvido para um mesmo conjunto de imagens contendo diversos objetos poliédricos.

Finalmente, foram projetadas, implementadas e avaliadas todas as etapas de um método para segmentação de imagens de profundidade com base em técnicas para detecção de bordas. Este método difere dos demais apresentados no trabalho de Hoover, principalmente, por se tratar de um segmentador baseado em detecção de bordas. As etapas do método com as suas principais contribuições podem ser resumidas por:

1. Filtragem da imagem por um filtro mediano. Foram identificados ruídos impulsivos que são satisfatoriamente suavizados por este filtro de tamanho 3x3, aplicado uma única vez na imagem.
2. Detecção de bordas de profundidade através do gradiente da banda (Z) da imagem de profundidade. Neste processo um limiar automático é calculado, baseado na média

¹<http://marathon.csee.usf.edu/range/seg-comp/SegComp.html>

e desvio padrão da imagem de gradiente, sugerido por Haddon [18]. Isto facilita a obtenção de um algoritmo genérico, que se adequa facilmente a tipos diferentes de imagens gerando resultados semelhantes. As bordas resultantes possuem apenas um *pixel* de espessura, facilitando os processos posteriores.

3. Cálculo dos vetores normais à superfície de cada *pixel* da imagem. Um processo de rotulação dos *pixels* válidos em uma janela de tamanho $N \times N$ é realizado para evitar que *pixels* que estejam em lados opostos a uma borda de profundidade sejam utilizados para o cálculo dos coeficientes do vetor normal, causando uma distorção nos valores dos vetores.
4. Detecção de bordas de orientação baseado no gradiente angular dos vetores normais e no cálculo de um limiar automático. Alguns problemas relacionados ao cálculo do vetor normal à superfície foram levantados e aprimorados pelo uso de uma abordagem mais apropriada.
5. Refinamento do mapa de bordas com a união das bordas de profundidade e orientação, afinamento e fechamento do mapa de bordas. O algoritmo de afinamento desenvolvido é um método híbrido totalmente independente de limiares, baseado no algoritmo de esqueletização e em Morfologia Matemática. Este algoritmo fornece uma borda de espessura de apenas um *pixel* para diferentes padrões de bordas. No fechamento, uma borda é traçada entre todos os *pixels* de extremidade e o *pixel* de borda próximo mais apropriado. Este processo garante um mapa de bordas fechado, obtendo-se a segmentação da imagem em regiões delimitadas por bordas.
6. Crescimento de regiões guiado pelo mapa de bordas com o objetivo de fornecer uma segmentação final mais precisa, ou seja, com as regiões mais significativas bem definidas. A eliminação das regiões muito pequenas baseada na orientação e profundidade de cada *pixel* da superfície, é realizada para preservar a topologia dos objetos e eliminar regiões de ruído. Esta etapa é de fundamental importância para fornecer uma segmentação mais confiável e adequada a um processo posterior de representação.

Como primeira contribuição do trabalho, ressaltamos que o método desenvolvido é totalmente modular. Assim, pode-se realizar o processo de forma paralela desde a primeira etapa, otimizando o tempo de execução do algoritmo e preservando ainda mais a topologia dos objetos da imagem. Isso é possível pois, ao serem identificadas as bordas de profundidade, um crescimento de regiões é realizado definindo algumas regiões da imagem. Dessa forma, as etapas seguintes são executadas em paralelo para todas as regiões identificadas inicialmente. No pior caso, pode existir apenas uma região na imagem de entrada, mesmo depois do crescimento de regiões no mapa de bordas de profundidade. Entretanto, para todas as imagens avaliadas nos experimentos, esta situação não ocorreu. Ao final do processamento de cada uma das regiões, obtém-se a segmentação final.

Outra contribuição deste trabalho está na preservação da forma e da topologia das superfícies dos objetos, que é mais evidenciada na aplicação do processamento paralelo para regiões distintas da imagem, ao invés do processamento da imagem toda. Isso ocorre pois, no processamento paralelo, os *pixels* das bordas de cada região delimitada não sofrem influência dos *pixels* das regiões vizinhas. Além disso, é possível agregar novas etapas ao método sem alterar suas etapas iniciais, como por exemplo, a segmentação de superfícies curvas aproximadas por planos, para imagens compostas por objetos não poliédricos. Para esta situação, a segmentação final gerada pelo método desenvolvido é formada por regiões planares e curvas. Então, para cada uma das regiões identificadas como regiões curvas, um novo processo é realizado em paralelo para segmentar estas superfícies, fornecendo uma segmentação mais apropriada, facilitando a sua representação e interpretação como apresentado no trabalho de Bellon et al. [5].

Como a grande maioria dos algoritmos de segmentação utilizam diversos limiares fixos, que dificilmente se adaptam para tipos diferentes de imagens, o desenvolvimento e aplicação de limiares automaticamente calculados também é uma contribuição deste trabalho. Os limiares automáticos são mais eficientes do ponto de vista da generalização, porém demandam mais tempo de processamento para serem obtidos. Muitas vezes o custo computacional para o cálculo de um limiar automático é compensado quando o processo é aplicado para uma grande variedade de imagens de entrada fornecendo resultados

semelhantes.

Tendo em vista que uma grande dificuldade dos métodos baseados em detecção de bordas está na localização e precisão das bordas, especialmente as bordas de orientação, uma contribuição vital desse trabalho está no desenvolvimento do método híbrido de afinamento, conforme é mostrado no decorrer deste documento.

Com o estudo dos resultados comparativos do método desenvolvido e dos algoritmos de segmentação de imagens de profundidade avaliados neste trabalho, foram identificados alguns problemas e apontadas possíveis soluções. Os resultados experimentais mostraram que o método desenvolvido é eficaz e eficiente para a segmentação de imagens de profundidade quando comparado aos demais segmentadores apresentados. Em alguns casos forneceu os melhores resultados com relação à topologia das superfícies e eliminação de ruídos, mostrando que a detecção de bordas aplicada à segmentação de imagens pode resolver muitos dos problemas identificados na segmentação de imagens de profundidade.

Este trabalho é composto por seis capítulos. O primeiro é a introdução, seguido do capítulo 2, onde são apresentados alguns conceitos básicos fundamentais para a compreensão da segmentação de imagens e dos principais processos que são abordados neste trabalho. No capítulo 3 são vistos os trabalhos relacionados à segmentação de imagens de profundidade e à comparação destes trabalhos. O método desenvolvido para segmentação baseado na detecção de bordas é apresentado no capítulo 4, sendo discutidas considerações com relação aos resultados obtidos em cada etapa. Os resultados experimentais do método desenvolvido e dos segmentadores comparados neste trabalho são mostrados no capítulo 5, seguido de uma conclusão no capítulo 6.

CAPÍTULO 2

CONCEITOS BÁSICOS

Este capítulo tem por objetivos apresentar alguns conceitos fundamentais para a compreensão da segmentação de imagens e descrever os principais processos que são mostrados na sequência deste trabalho.

2.1 Visão humana e visão computacional

A visão humana é formada por um complexo sistema biológico que permite aos seres humanos interpretar radiações eletromagnéticas capturadas pelos olhos. Estas radiações, vindas da reflexão da luz emitida nas superfícies dos objetos da cena observada, dão origem a sinais neuronais que são processados e armazenados em diversas áreas do cérebro.

A informação visual, ou seja a imagem, é processada em paralelo, em diversos níveis e em diversas áreas do cérebro especializadas na identificação de uma determinada característica desta imagem, por exemplo as bordas dos objetos, a profundidade, a forma, entre outras [29].

As características obtidas nos diversos níveis de processamento, quando comparadas às informações anteriormente adquiridas pelo ser humano, fornecem a interpretação e o reconhecimento do que está sendo visto ou analisado, podendo ainda gerar uma resposta do organismo ao estímulo visual.

A estrutura fisiológica do olho humano é capaz de manter uma adaptação constante e rápida em relação a mudanças no ambiente, como por exemplo a intensidade de luz e cores. Estas capacidades são muitas vezes difíceis de serem empregadas em sistemas artificiais, como por exemplo um sistema capaz de reconhecer um objeto em diferentes ambientes, tais como, sobre chuva ou pouca luminosidade.

A percepção de cores pelo olho humano baseia-se na hipótese formulada por Young em 1801, que estabelece que os cones (células fotossensíveis que compõem a retina juntamente

com os bastonetes) se subdividem em três classes, com diferentes sensibilidades situados em torno do vermelho, do verde e do azul. Desta forma, todas as sensações de cor percebidas pelo olho humano são na verdade combinações das intensidades dos estímulos recebidos por cada um destes tipos de cones [33].

Os sistemas de visão computacional visam executar de forma rápida, eficiente e confiável, as mesmas tarefas do sistema de visão humano. Porém, o atual conhecimento científico e tecnológico não permite a criação de sistemas que tenham um desempenho equivalente ao sistema humano, impondo uma série de restrições quanto ao tempo de resposta necessário a muitas aplicações e a qualidade dos resultados obtidos.

Em algumas aplicações, os sistemas de visão computacional são mais eficientes que o sistema de visão humano, como por exemplo no reconhecimento de padrões para um sistema de inspeção de falhas em circuitos impressos. Neste exemplo a rapidez da máquina e a confiabilidade nas respostas podem ser na grande maioria das vezes melhor processada do que o olho humano. Sem levar em consideração o tempo, que também é bem menor pois, o processo pode ser contínuo e não sofre fadiga como o olho humano.

A complexidade dos processos de visão humana vem sendo desvendada, reproduzida e aplicada em pequenas etapas nos sistemas de visão computacional. Os grandes avanços nesta área de pesquisa, vêm desenvolvendo inúmeros sistemas de visão para senão substituir a visão humana, auxiliá-la em processos de precisão e confiabilidade em aplicações que demandem um tempo de resposta mais rápido que o do ser humano.

2.2 Imagens de intensidade luminosa

As imagens de intensidade luminosa são as imagens que diariamente são processadas pelos nossos olhos. Para efeito de exemplificação, considere que está sendo utilizada uma imagem de intensidade luminosa monocromática, também chamada de imagem de tons de cinza. Neste caso, uma imagem de intensidade luminosa pode ser descrita matematicamente [17] por:

$$f(x, y) = i(x, y) \times r(x, y) \quad (2.1)$$

Onde:

$$\begin{aligned} i(x, y) & \text{ é a intensidade de luz incidente no ponto } (x, y) \\ r(x, y) & \text{ é a reflectância no ponto } (x, y) \end{aligned} \quad (2.2)$$

Com:

$$\begin{aligned} 0 < i(x, y) < \infty \\ 0 < r(x, y) < 1 \end{aligned} \quad (2.3)$$

É possível observar que a função $f(x, y)$ tem um domínio infinito, e portanto, para representá-la e poder processá-la em um computador, é necessário fazer uma discretização espacial e outra em amplitude.

A amostragem é o processo de discretização espacial, que converte a imagem em uma matriz de $M \times N$ pontos, cada qual denominado *pixel*:

$$f(x, y) = \begin{bmatrix} f(0, 0) & f(0, 1) & \dots & f(0, N - 1) \\ f(1, 0) & f(1, 1) & \dots & f(1, N - 1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M - 1, 0) & f(M - 1, 1) & \dots & f(M - 1, N - 1) \end{bmatrix} \quad (2.4)$$

O tamanho desta matriz está diretamente ligado à resolução (qualidade) da imagem. Quanto mais pontos forem amostrados, melhor a imagem representará a cena, entretanto maior será o tempo de processamento desta imagem. Em um sistema de visão computacional é preciso determinar a relação aceitável entre qualidade dos resultados e tempo de processamento, para uma determinada aplicação. Se a aplicação está voltada à obtenção de resultados com maior precisão e qualidade, o tempo será sacrificado, porém se a velocidade é a característica mais importante, então a resolução da imagem pode ser reduzida.

A discretização em amplitude é dada pelo processo de quantização da imagem, onde cada *pixel* da matriz assume um valor inteiro, na faixa de 0 a $2^n - 1$. Quanto maior o valor de n , maior o número de níveis, ou de tons de cinza, dessa imagem.

Por convenção, os tons de cinza mais claros na imagem são os valores maiores na matriz, e os valores menores da matriz são os tons mais escuros na imagem. Normalmente adota-se valores entre 0 e 255, ou seja $n = 8$ que fornece 256 tons diferentes para uma imagem de tons de cinza.

Nas imagens coloridas que possuem informações em intervalos ou bandas distintas de frequência, é necessário uma função $f(x, y)$ para cada banda. Por exemplo, as imagens coloridas no padrão RGB, são formadas por cores primárias aditivas: vermelho (R - *red*), verde (G - *green*) e azul (B - *blue*), como a sensibilidade dos três tipos cones do olho humano descritas na seção 2.1. Neste caso a representação da imagem é dada por uma matriz de $M \times N$, onde cada ponto da matriz é formado por um vetor de 3 valores, um para a intensidade de cada cor, formando assim uma imagem de 3 bandas (R,G,B). A imagem de intensidade luminosa monocromática é formada por uma única banda (Tons de cinza).

As imagens de intensidade luminosa são obtidas principalmente por meios ópticos como uma câmera ou *scanner* [16]. A maior deficiência dessas imagens é a ausência da informação de profundidade, perdida na projeção 3D-2D [17], o que dificulta sua interpretação.

Para superar essa deficiência, ao longo dos últimos 30 anos foram desenvolvidos vários métodos para recuperar a informação 3D [6]. Essa busca culminou com o advento das imagens de profundidade, cujas características são apresentadas a seguir.

2.3 Imagens de profundidade

As imagens de profundidade, chamadas *range images*, têm sido cada vez mais utilizadas em processamento de imagens e visão computacional. Sua grande vantagem em relação às imagens de intensidade luminosa é que fornecem as informações tridimensionais (3D) da cena observada de forma explícita. Permitindo assim, extrair diversas características importantes para a interpretação da imagem como, por exemplo, classificação de bordas

e a orientação das superfícies.

Nelas, cada *pixel* da imagem é representado por um um vetor com 3 valores em uma matriz $M \times N$. Estes valores significam o conjunto de coordenadas tridimensionais (x, y, z) do *pixel*, ou seja, estas imagens são formada por 3 bandas (X, Y, Z) .

Estas imagens são obtidas através de métodos como a triangulação ou métodos baseados em tempo de reflexão. Estes métodos são utilizados em equipamentos compostos por um sistema de ultra-som ou raios *laser* e também em equipamentos compostos por câmeras CCD (*charge coupled device*) [6, 23]. Estes equipamentos permitem a obtenção da distância de cada *pixel* da superfície dos objetos observados até a origem de um sistema de coordenadas [3, 6, 23].

Existem atualmente diversos equipamentos que fazem a aquisição de imagens de profundidade. Cada um deles com uma característica específica na forma de obtenção e no formato de armazenamento dos dados obtidos. Nas subseções seguintes são apresentados 2 tipos diferentes de imagens de profundidade (ABW e Perceptron), obtidas por métodos diferentes.

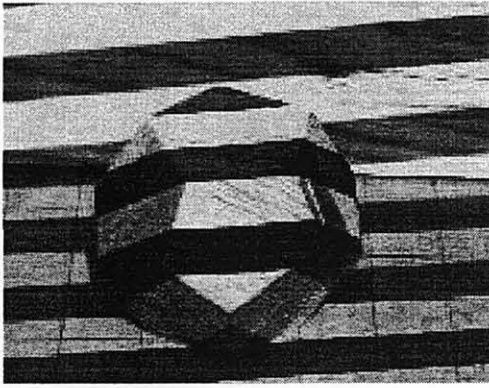
2.3.1 ABW

No sensor ABW a imagem de profundidade é obtida por um *scanner* de luz estruturada (*structured light scanner*), onde a profundidade dos *pixels* da imagem é dada através da análise de feixes de luz estruturada sobre os objetos.

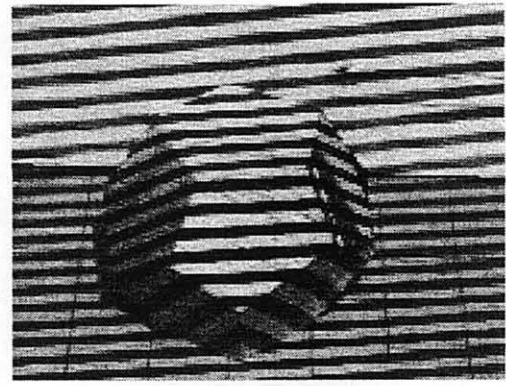
O equipamento é formado por uma câmera CCD e um projetor de luz que emite feixes com diferentes padrões e larguras. Este equipamento tem a vantagem de utilizar uma técnica relativamente rápida e com baixo custo financeiro, sendo utilizada por vários grupos de pesquisa atualmente [40].

O processo de aquisição dá-se a partir da iluminação dos objetos da cena por diferentes feixes de luz, que produzem diferentes imagens como mostra a figura 2.1. Estas imagens são analisadas e, por triangulação, é possível obter a distância de profundidade de cada *pixel* da superfície dos objetos da cena observada.

A figura 2.2 mostra o processo de aquisição da informação de profundidade do objeto.



(a) Imagem 1



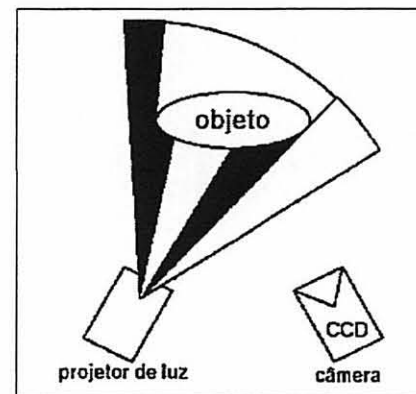
(b) Imagem 2

Figura 2.1: Objeto iluminado por feixes de luz com diferentes padrões.

Neste exemplo, se um *pixel* aparece escuro na Imagem 1 (figura 2.2(a)) e claro na Imagem 2 (figura 2.2(b)), isso indica que o *pixel* do objeto está entre o segundo quarto do campo de vista do projetor de luz. Sabendo-se a distância entre o projetor e a câmera e o ângulo de projeção de cada feixe, por triangulação obtém-se a informação de profundidade do *pixel*.



(a) Imagem 1



(b) Imagem 2

Figura 2.2: Processo de aquisição da informação de profundidade dos objetos.

Este equipamento utiliza a projeção ortogonal, de forma que o sistema de coordenadas da imagem é equivalente ao sistema de coordenadas Cartesianas (X, Y, Z). Para a obtenção dos valores (X, Y, Z) de cada *pixel* da imagem é necessário apenas mudanças no fator de escala dos valores obtidos na aquisição.

Para cada *pixel* $r(i, j)$ na imagem de profundidade ABW:

$$0 \leq i, j \leq 511 \quad (2.5)$$

Onde o índice das linhas da imagem de profundidade são indicados por i e as colunas por j . A conversão para as coordenadas Cartesianas com precisão em milímetros é dada por:

$$\begin{aligned} x[i, j] &= (j - 255) \cdot \frac{\left(\frac{r[i, j]}{scal + offset} \right)}{|f_k|} \\ y[i, j] &= \frac{(255 - i)}{c} \cdot \frac{\left(\frac{r[i, j]}{scal + offset} \right)}{|f_k|} \\ z[i, j] &= \frac{1}{scal} \cdot (255 - r[i, j]) \end{aligned} \quad (2.6)$$

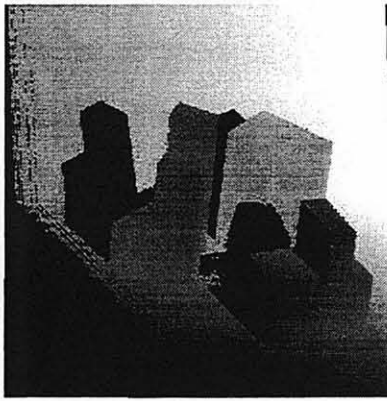
Os valores $offset$, $scal$, f_k e c são obtidos pelo processo de calibração do equipamento.

Por exemplo:

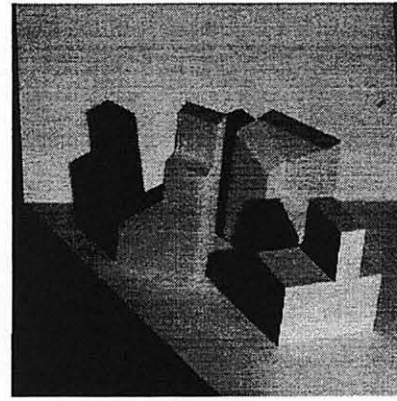
$$\begin{aligned} (1) \quad & offset = 785.4, scal = 0.774, f_k = -1611.0, c = 1.45 \\ (2) \quad & offset = 771.0, scal = 0.792, f_k = -1586.1, c = 1.45 \end{aligned} \quad (2.7)$$

Uma característica identificada nestas imagens é a presença de ruídos ou ausência de informação em diversas regiões da imagem. As regiões sem informação são relativas a sombras projetadas pelos objetos iluminados como mostra a figura 2.3. A figura 2.3(a) é a imagem de profundidade e a figura 2.3(b) mostra a imagem de intensidade luminosa da mesma cena.

Estas imagens de profundidade têm o tamanho de 512x512 com precisão de até 256 valores de profundidade, ou seja 8 bits.



(a) Imagem de profundidade



(b) Imagem de intensidade

Figura 2.3: *Imagem do tipo ABW.*

2.3.2 Perceptron

As imagens do sensor Perceptron são capturadas por um *scanner laser*, chamado *laser range finder*. Este tipo de equipamento calcula a profundidade baseado no tempo e na amplitude do sinal refletido de cada pulso de raio *laser*. O tempo de reflexão gera a imagem de profundidade. Ela é obtida pelo deslocamento da direção do raio em incrementos angulares pequenos de maneira a capturar a informação de profundidade de todos os pontos da cena observada.

A amplitude do sinal refletido gera uma imagem de reflectância. Esta imagem de reflectância é uma pseudo-imagem de intensidade luminosa, porém sem a presença de sombras.

Estas imagens têm tamanho 512x512 com precisão de até 4092 valores de profundidade, ou seja 12 bits. Como o equipamento utiliza uma projeção perspectiva, o sistema de coordenadas da imagem é esférico. Neste caso, a conversão da imagem de profundidade para um sistema de coordenadas Cartesianas (X, Y, Z) é feita com algumas equações de transformações.

Para cada *pixel* $r(i, j)$ na imagem de profundidade Perceptron:

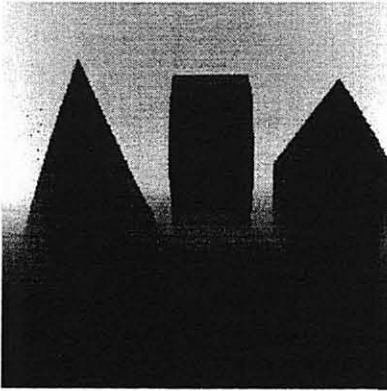
$$0 \leq i, j \leq 511 \quad (2.8)$$

Onde o índice das linhas da imagem de profundidade são indicados por i e as colunas por j . A conversão para as coordenadas Cartesianas com precisão em centímetros é dada por:

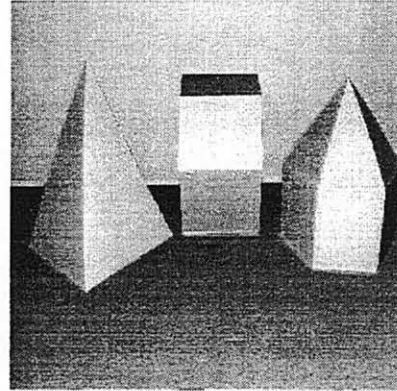
$$\begin{aligned}
 x[i, j] &= dx + r_3 \cdot \sin(\alpha) & \alpha &= \alpha_0 + H \frac{(255.5 - j)}{512.0} \\
 y[i, j] &= dy + r_3 \cdot \cos(\alpha) \sin(\beta) & \beta &= \beta_0 + V \frac{(255.5 - i)}{512.0} \\
 z[i, j] &= dz + r_3 \cdot \cos(\alpha) \cos(\beta) \\
 r_1 &= \frac{(dz - h_2)}{\delta} & dx &= (h_2 + dy) \cdot \tan(\alpha) \\
 r_2 &= \sqrt{\frac{(dx)^2 + (h_2 + dy)^2}{\delta^2}} & dy &= dz \cdot \tan(\theta + \frac{1}{2}\beta) \\
 r_3 &= (r[i, j] + r_0 - (r_1 + r_2)) \cdot \delta & dz &= -h_1 \cdot \frac{(1.0 - \cos(\alpha))}{\tan(\gamma)}
 \end{aligned} \tag{2.9}$$

Os valores h_1 , h_2 , γ , θ , α_0 , β_0 , H , V , r_0 e δ são obtidos pelo processo de calibração do equipamento.

A figura 2.4 mostra um exemplo de uma imagem de profundidade do tipo Perceptron. A figura 2.4(a) é a imagem de profundidade e a figura 2.4(b) mostra a imagem de reflectância da mesma cena.



(a) Imagem de profundidade



(b) Imagem de reflectância

Figura 2.4: Imagem do tipo Perceptron.

Tanto as imagens ABW quanto as Perceptron foram utilizadas no processo de comparação dos segmentadores de imagens de profundidade no trabalho de Hoover et al. [21].

2.4 Imagens binárias

As imagens binárias podem ser representadas como imagens de intensidade luminosa monocromática, cujo processo de quantização da imagem fornece somente valores inteiros 0(zero) ou 1(um). Ou seja, estas imagens são imagens de 1 bit. Por convenção os *pixels* com valor 0 são os pontos de fundo da imagem e os *pixels* com valor 1 são *pixels* que possuem a característica analisada de um determinado processo.

Estas imagens são utilizadas como imagens resultantes de processos como, por exemplo, a detecção de bordas. Nelas podemos identificar os *pixels* de borda dos objetos com valor 1 e o restante dos *pixels* da imagem com valor 0.

2.5 Máscaras

Um processo sobre uma imagem pode ser realizado de três formas:

- 1) Pontualmente: o valor do *pixel* $p_r(x, y)$ da imagem resultante é calculada baseado somente no valor do *pixel* $p_o(x, y)$ da imagem original.
- 2) Por máscaras: o valor do *pixel* $p_r(x, y)$ da imagem resultante é calculado baseado no valor do *pixel* $p_o(x, y)$ da imagem original e no valor dos *pixels* vizinhos que estão dentro da máscara.
- 3) Pela análise global: O valor do *pixel* $p_r(x, y)$ da imagem resultante é calculado a partir da análise global de uma característica. Por exemplo, pela análise do histograma da imagem, é possível identificar se a imagem está escura e equalizar a imagem deixando-a mais clara, baseando-se nesta informação global do histograma.

A grande maioria do processamento de imagens é realizado utilizando máscaras. As máscaras são pequenas matrizes quadradas (normalmente de tamanho 3x3, 5x5, 7x7, ...)

que podem ser aplicadas a cada *pixel* da imagem, centrado na máscara.

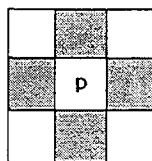
Quando aplicamos uma máscara a uma imagem, ela opera sobre o *pixel* da imagem que está exatamente sob o seu elemento central, utilizando informações dos *pixels* vizinhos. Ou seja, o novo valor de cada *pixel* da imagem será a aplicação da máscara centrada neste *pixel*.

É desta forma, por exemplo, que funciona o processo de suavização ou filtragem de ruídos. Na filtragem, os *pixels* com valores discrepantes em relação a seus vizinhos têm seus valores homogeneizados.

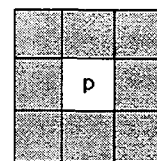
Se considerarmos as máscaras apenas como delimitadores para um determinado processo, podemos dizer que estamos utilizando uma janela. Somente os *pixels* da imagem sob esta janela é que são levados em consideração no processo, por exemplo no cálculo de erro de ajuste de um plano. Neste processo somente os *pixels* da janela são utilizados no cálculo.

2.5.1 Vizinhança

A vizinhança de um *pixel* $p(x, y)$ é definida como o conjunto dos *pixels* que ficam ao seu redor. Entre os tipos de vizinhança existentes, dois deles são os mais empregados: vizinhança 4 e vizinhança 8. A vizinhança 4 considera como vizinhos apenas os *pixels* que estão ao redor do *pixel* $p(x, y)$ na direção vertical e horizontal, enquanto que na vizinhança 8 também são considerados vizinhos os *pixels* que estão nas diagonais. A figura 2.5 ilustra este conceito de vizinhança, onde os *pixels* cinza representam os vizinhos do ponto p segundo cada tipo de vizinhança.



(a) Vizinhança 4



(b) Vizinhança 8

Figura 2.5: Tipos de vizinhança mais utilizados (vizinhança 4 e 8).

Dependendo da fase de um processo, pode ser necessário o uso de um tipo específico

de vizinhança, como por exemplo o processo de afinamento de bordas.

2.5.2 Filtros

Os filtros são máscaras, cujos *pixels* componentes possuem valores que definem a utilidade do filtro, por exemplo um filtro para suavização de ruídos, um filtro de detecção de bordas, etc.

Num processo de filtragem, o processamento de um *pixel* $p(x, y)$ depende dos valores desse *pixel* e seus vizinhos. Em geral, na vizinhança, os *pixels* mais próximos de $p(x, y)$ contribuem mais na definição do novo valor de $p(x, y)$ do que os *pixels* mais afastados.

O principal objetivo da filtragem é processar uma certa imagem de modo que a imagem resultante seja mais adequada que a imagem original, para uma aplicação específica. Muitas vezes os filtros podem gerar imagens de outros domínios, como é o caso dos filtros que atuam no domínio da frequência (transformada de Fourier), cuja imagem resultante nada se parece com a imagem original [33].

Num filtro mediano, o *pixel* da imagem centrado na máscara é substituído pela mediana dos *pixels* situados em sua vizinhança. A mediana m de um conjunto de n elementos é o valor tal que metade dos n elementos do conjunto situam-se abaixo de m e a outra metade acima de m . Quando n é ímpar, a mediana é o próprio elemento central do conjunto ordenado. Nos casos em que n é par, a mediana pode ser calculada pela média aritmética dos dois elementos mais próximos do centro. Neste processo, a ordenação (*sorting*) constitui uma etapa de tempo de processamento relativamente alto, apesar de inúmeros métodos eficientes existentes na literatura, como por exemplo o filtro *pseudo-median* sugerido por Pratt [33].

Um filtro mediano tem a função de remover ruídos em imagens, preservando as bordas e detalhes finos, principalmente em situações nas quais a imagem é contaminada por ruídos impulsivos “sal-e-pimenta”. Em situações em que o ruído é do tipo Gaussiano, seu desempenho é apenas satisfatório, comparável ao do filtro média. O filtro média é o filtro onde o valor do *pixel* central da máscara recebe o valor da média de seus vizinhos.

A figura 2.6 mostra o efeito de um filtro mediano e de um filtro média, ambos de



(a) Imagem original

(b) Filtro média

(c) Filtro mediano

Figura 2.6: *Processo de filtragem pelo filtro média e filtro mediano em uma imagem de profundidade ABW.*

tamanho 7x7, quando aplicado em uma imagem de profundidade com ruídos impulsivos.

As imagens ABW possuem ruídos de diversos níveis que podem ser satisfatoriamente filtrados por um filtro mediano.

A grande dificuldade no processo de filtragem é conseguir uma boa relação no tamanho da máscara para o filtro e o número de vezes em que o filtro deve ser aplicado, de forma a não comprometer as características da imagem e conseguir eliminar satisfatoriamente os ruídos.

2.6 Morfologia Matemática

A Morfologia Matemática vem sendo aplicada nos últimos anos para resolver diversos problemas de processamento de imagens. Ela permite processar imagens com objetivos [38]: realce, segmentação, detecção de bordas, esqueletização, afinamento, etc.

O princípio de base da Morfologia Matemática consiste em extrair uma informação relativa à geometria e à topologia de um conjunto desconhecido de uma imagem pela transformação através de um outro conjunto completamente definido chamado elemento estruturante.

Os elementos estruturantes são máscaras normalmente de tamanho 3x3 com valores convencionados como: 0(zero) para *pixel* marcados e 1(um) os não marcados. Em al-

guns casos um elemento estruturante é aplicado em uma operação considerando suas 4 variações. Estas variações são dadas pelas rotações (0, 90, 180 e 270 graus) desta máscara, de maneira a cobrir todas as possíveis configurações dadas por este elemento.

A figura 2.7 mostra exemplos de elementos estruturantes. E a figura 2.8 mostra um elemento estruturante e suas variações rotacionadas, onde os *pixels* marcados com *x* não são avaliados na operação deste elemento estruturante, pois podem assumir qualquer valor. Por exemplo, no caso do afinamento das bordas, estes *pixels* marcados com *x* podem ou não ser de bordas, que a operação terá o mesmo resultado.

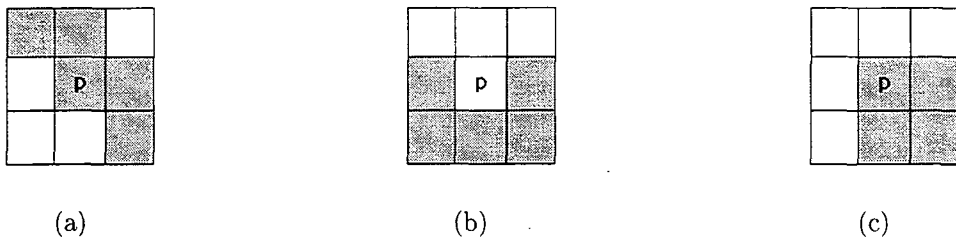


Figura 2.7: Exemplo de 3 elementos estruturantes.

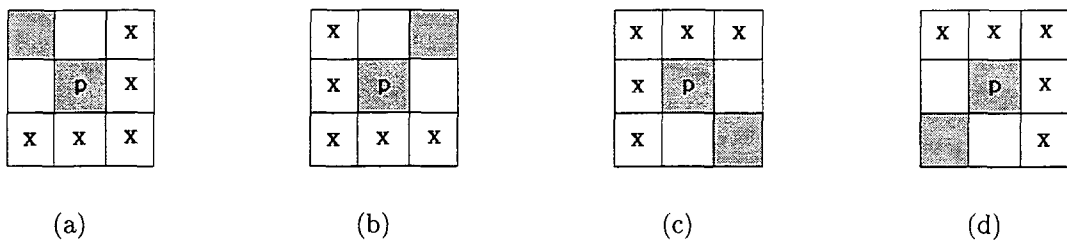


Figura 2.8: Exemplo das 4 variações de um elemento estruturante (0, 90, 180 e 270 graus).

De forma geral, a Morfologia Matemática aplica-se principalmente a imagens binárias, mas também existem operadores 3D aplicados a imagens tridimensionais. Cada *pixel* $p(x, y)$ da imagem e sua vizinhança são analisados. Se uma configuração de pontos marcados e não marcados equivalentes a um determinado elemento estruturante é encontrada em uma região da imagem, o *pixel* equivalente na imagem resultante, centrado sob o elemento estruturante, recebe um valor que o caracteriza segundo a operação morfológica. Por exemplo, no processo de afinamento de uma imagem binária de bordas, os *pixels* onde é encontrada a configuração de um elemento estruturante são eliminados na imagem

resultante de maneira a afinar o mapa de bordas. A operação utilizada neste caso é a erosão.

Existem diversos operadores morfológicos que podem ser utilizados: erosão, dilatação, abertura, fechamento, poda, entre outros [38].

2.7 Gradientes

O gradiente é a medida de mudança numa função, e uma imagem pode ser considerada como sendo um conjunto de amostras de uma função contínua, descrita na seção 2.1.

O gradiente de uma imagem é o equivalente à derivada de duas direções e pode ser definido como:

$$G[f(x, y)] = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\delta f}{\delta x} \\ \frac{\delta f}{\delta y} \end{bmatrix} \quad (2.10)$$

A magnitude do gradiente é dada por:

$$M[f(x, y)] = \sqrt{G_x^2 + G_y^2} \quad (2.11)$$

E a direção do gradiente é definida como:

$$\alpha(x, y) = \tan^{-1} \left(\frac{G_y}{G_x} \right) \quad (2.12)$$

Para imagens digitais, as derivadas na equação 2.10 são aproximadas por diferenças entre o valor de um *pixel* e seus vizinhos. Uma aproximação simples de gradiente é dada por:

$$\begin{aligned} G_x &\approx |f(x, y+1) - f(x, y-1)| \\ G_y &\approx |f(x+1, y) - f(x-1, y)| \end{aligned} \quad (2.13)$$

No caso de imagens de intensidade luminosa, o gradiente determina a mudança de

intensidade de cada *pixel* em relação aos seus vizinhos. Nas imagens de profundidade, utilizando-se a banda (Z), o gradiente representa a mudança de profundidade entre os *pixels*.

2.8 Bordas

Bordas são descontinuidades na amplitude de uma determinada característica da imagem. Elas são de fundamental importância no processamento de imagens, pois podem fornecer uma descrição topológica aproximada dos objetos da imagem e suas superfícies. Pode-se dizer que os *pixels* que estão na fronteira entre duas regiões com características relativamente distintas são considerados como bordas.

Nas imagens de profundidade, pode-se identificar diversos tipos de bordas, sendo as de maior aplicação as bordas de profundidade (*step edges*) e as bordas de orientação (*roof edges*) [20]. As bordas de profundidade são compostas por *pixels* que estão na interseção entre regiões com valores de profundidade significativamente distintos, enquanto que as bordas de orientação estão localizadas no encontro de superfícies com orientações distintas.

A figura 2.9 mostra a diferença entre a borda de profundidade e a borda de orientação de uma imagem de profundidade.

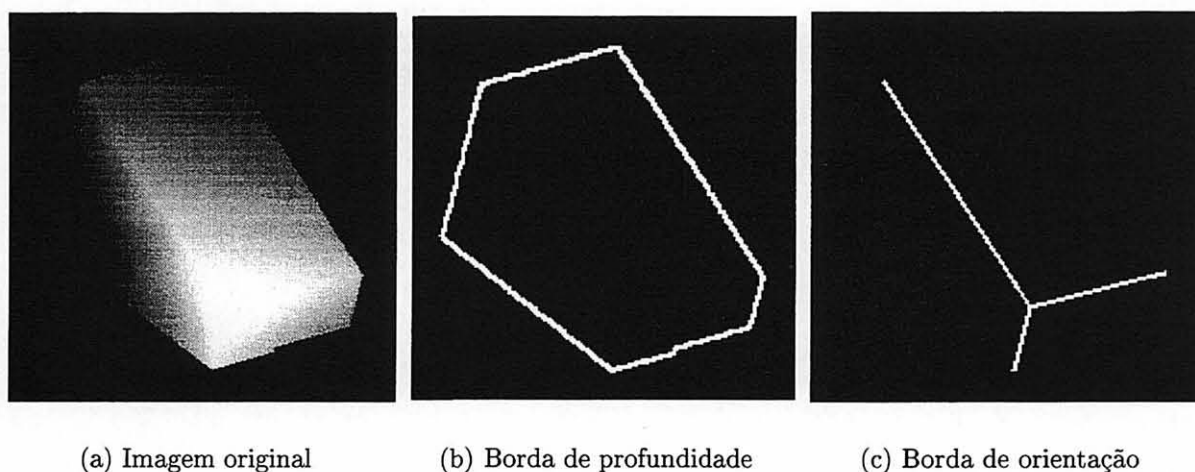


Figura 2.9: *Diferentes tipos de bordas de uma imagem de profundidade.*

Em linhas gerais, para determinar se um *pixel* da imagem é ou não de borda, calcula-se o gradiente deste *pixel* (seção 2.7) e, caso o gradiente seja maior do que um valor de limiar

pré-definido, o *pixel* é considerado como borda.

Para detetar a borda de profundidade, é necessário somente a informação da banda (Z) da imagem que fornece a profundidade para o cálculo do gradiente. Para a borda de orientação são necessárias as três bandas (X, Y, Z) para o cálculo da orientação das superfícies, que fornece o gradiente angular de cada *pixel*.

2.9 Limiares

Limiares (*threshold*) são valores utilizados em diversas fases do processamento de imagens. Estes valores determinam ou limitam a ação de um determinado procedimento.

A função do limiar consiste em fornecer um limite máximo ou mínimo que uma determinada característica da imagem deve assumir para que esta seja identificada na imagem. Em outras palavras, é um valor de corte, os valores que estiverem abaixo ou acima deste limiar são tratados diferentemente.

No caso de um detetor de bordas de profundidade, é necessário determinar um limiar T para identificar os *pixels* de borda da imagem. Se um *pixel* $p_1(x, y)$, por exemplo, tiver na sua vizinhança um *pixel* $p_2(x, y)$ com valor muito maior, ou muito menor que o seu, então $p_1(x, y)$ pode ser um possível candidato a *pixel* de borda. Para determinar o que é o “muito maior” ou o “muito menor”, utiliza-se os limiares. Neste exemplo, se considerarmos que o limiar $T = 50$, todos os *pixels* que tiverem a diferença de seu valor com o valor de um de seus vizinhos maior que 50, são marcados como *pixels* de borda na imagem resultante.

O grande problema está em definir um valor de limiar ideal para um determinado grupo de imagens. Muitas vezes, um limiar que fornece um bom resultado para uma imagem não produz resultados semelhantes para outra, principalmente quando trabalha-se com diferentes bases de dados de imagens [37].

Em sistemas de visão computacional, existe a necessidade de se definir um conjunto de limiares adequados para uma determinada aplicação. Entretanto, na fase de treinamento dos algoritmos, as imagens utilizadas e até mesmo o ambiente onde o sistema está sendo treinado têm uma determinada característica. Quando este mesmo sistema é colocado

em outro ambiente com características diferentes, muitas vezes os limiares não fornecem resultados semelhantes. Desta forma, freqüentemente o tempo computacional é sacrificado para que um limiar seja mais adequadamente calculado, e se possível de forma automática. Por exemplo, no uso de visão em robôs autônomos, é necessário que os limiares dos algoritmos sejam calculados de forma automática para que o robô tenha desempenho equivalente independente do ambiente onde é testado.

Neste trabalho, foram identificados alguns problemas no uso de limiares fixos e limiares automaticamente calculados para diferentes tipos de imagens de entrada. Muitas vezes o custo computacional para o cálculo de um limiar automático é compensado quando o processo é aplicado para uma grande variedade de imagens de entrada.

O trabalho de Haddon [18] é uma boa referência do estudo de abordagens para a determinação de valores de limiar em técnicas de detecção de bordas. Ele apresenta técnicas genéricas para a seleção de limiar baseando-se em estatísticas (média e variância) de ruído da imagem. Também descreve algumas técnicas eficientes para o cálculo de limiar que podem ser adotadas para resolver este problema.

Segundo Haddon, a seleção do limiar é tradicionalmente feita em uma de quatro maneiras:

- 1) O limiar é escolhido interativamente pelo usuário com base na observação da imagem;
- 2) O valor de limiar é $\pm n$ vezes o desvio padrão mais a média do gradiente da imagem;
- 3) O limiar é escolhido de forma que uma determinada percentagem dos *pixels* é marcada como sendo de borda (normalmente 5% dos *pixels* da imagem);
- 4) Em imagens em que a distribuição das bordas é multimodal, o limiar é escolhido como sendo um valor entre os picos.

Rothwell [35] utiliza um método que calcula uma imagem de limiares. Esta imagem fornece um limiar ideal para cada *pixel* da imagem, de forma que a detecção de bordas

seja mais eficiente, principalmente por preservar a topologia do objeto nas junções das bordas. Este cuidado é devido ao fato de que, quando é feita uma filtragem na imagem inicial por filtros Gaussianos, os detetores de bordas normalmente não identificam pontos que estão nas junções entre bordas de profundidade e bordas de orientação. Apesar desse ponto positivo, o trabalho de Rothwell falha por ser extremamente sensível a ruído.

2.10 Segmentação de imagens

Uma das maiores dificuldades em processamento de imagens é a segmentação de imagens, principalmente quando se trata de processos ou algoritmos genéricos que precisam manter resultados satisfatórios para uma determinada aplicação. Por exemplo, mísseis autônomos teleguiados, que precisam identificar o alvo após seu lançamento.

A segmentação, por definição, é o processo de dividir a imagem em regiões conexas baseado em uma ou mais característica da imagem. Em geral, pretende-se obter regiões significativas, que são de fundamental importância em processos posteriores, como por exemplo a representação e reconhecimento dos objetos dessa imagem.

A grande maioria dos métodos de segmentação utilizam uma única característica para o processo de segmentação, entretanto a confiabilidade de cada região muitas vezes necessita que outra característica da imagem seja considerada no processo.

As duas principais abordagens de segmentação são: a segmentação por regiões e a segmentação por contornos. A segmentação por contornos é baseada em algoritmos de detecção de bordas (*edge detection*) e a segmentação por regiões utiliza, principalmente, algoritmos de crescimento de regiões *region growing*, algoritmos de divisão-e-união (*split-and-merge*) e métodos de agrupamento (*clustering*). As seções seguintes descrevem alguns desses métodos de segmentação.

2.10.1 Detecção de bordas

Os estágios iniciais do processamento de imagens procuram identificar características que sejam relevantes para estimar a estrutura e propriedades dos objetos numa cena. Bordas

são uma dessas características e a detecção de bordas pode ser considerada como um primeiro passo na recuperação de informações de imagens.

Os algoritmos para detecção de bordas, em geral, contêm três passos: a filtragem, a salientação e a detecção. Alguns deles têm ainda uma quarta etapa de refinamento ou localização.

A filtragem é fundamental para reduzir o impacto do ruído na detecção. Como descreve a seção 2.5.2, a grande dificuldade é saber o tamanho da máscara, o tipo e quantas vezes este filtro será aplicado na imagem. Quanto maior a máscara do filtro, mais as bordas são espalhadas, com isso a espessura da borda aumenta e dificulta sua exata localização. Entretanto, se o filtro for de tamanho muito pequeno, o efeito do ruído é maior na borda.

Na salientação, para facilitar a localização da bordas, é essencial determinar mudanças de intensidade na vizinhança de um *pixel*. Neste passo, *pixels* com mudanças significativas de intensidade na sua vizinhança são ressaltados, geralmente através do cálculo da magnitude do gradiente.

No passo da detecção, nem todos os *pixels* que têm o gradiente diferente de 0(zero) são *pixels* de borda. Por isso, geralmente utilizam-se técnicas de limiar para determinar quais *pixels* são e quais não são *pixels* de borda.

Muitos detetores de borda incluem ainda um quarto passo, a localização. A localização da borda pode ser refinada a uma resolução *subpixel* e sua orientação pode ser determinada. Em muitos casos, a borda tem a espessura maior que um *pixel*, precisando ser afinada para fornecer um mapa de bordas mais adequado.

O problema de detetar bordas em imagens de profundidade tem sido bastante estudado nos últimos anos, e diversas abordagens foram propostas. Alguns trabalhos recentes fazem um levantamento e comparação de diversos métodos de detecção de bordas [19, 39].

Mintz [30] sugere um método estatístico para detetar bordas de profundidade e de orientação, em imagens de profundidade, em que um *pixel* é declarado de borda somente se há um consenso entre os vários processos do método. Os processos aplicados tentam determinar se o *pixel* está numa região de descontinuidade (de profundidade ou de orientação). Para tanto, são utilizados métodos de estimativa robusta para determinar um

ajuste local às janelas na vizinhança do *pixel* e acumular votos para cada janela. Esse método inova por não ser baseado no uso de filtros ou valores de limiar.

Seguindo a mesma abordagem, Mitiche e Aggarwal [31] utilizam modelos probabilísticos para identificar bordas em imagens de profundidade. O objetivo é que o método seja imune à existência de ruídos na imagem. Basicamente, este procedimento determina o melhor particionamento da vizinhança de cada *pixel* em duas regiões contíguas. Planos são ajustados a estas regiões e o erro de ajuste é calculado, permitindo a escolha da "melhor" partição a ser utilizada na extração das bordas.

As subseções seguintes descrevem alguns algoritmos genéricos para a detecção de bordas, estudados durante este trabalho que reúnem um conjunto de características e métodos interessantes que os diferenciam.

2.10.1.1 Canny

O detetor de bordas desenvolvido por Canny [9] é considerado o método padrão utilizado por muitos pesquisadores. Seu funcionamento basicamente é composto de duas etapas.

Na primeira etapa, utiliza um filtro Gaussiano para suavizar ruídos e então calcula a intensidade da borda e sua direção em cada *pixel* da imagem suavizada. Isso é realizado, diferenciando a imagem em duas direções ortogonais e calculando a magnitude e a orientação do gradiente.

Para a imagem $I(x, y)$, o resultado da aplicação de um filtro de suavização Gaussiano $G[x, y; \sigma]$ sobre a imagem é uma imagem suavizada $S(x, y)$, dada por:

$$S(x, y) = G[i, j; \sigma] \otimes I(x, y) \quad (2.14)$$

onde σ determina o tamanho do filtro Gaussiano e controla o grau de suavização.

O gradiente da imagem suavizada $S(x, y)$ é calculado de forma semelhante à apresentada na seção 2.7.

A magnitude $M(x, y)$ e a orientação do gradiente $\theta(x, y)$ podem ser calculadas através das fórmulas 2.11 e 2.12 respectivamente, apresentadas na seção 2.7.

A segunda etapa se caracteriza por identificar quais são os *pixels* candidatos a borda. Estes *pixels* são identificados como os restantes de um processo de afinamento chamado *non-maximal suppression* que será apresentado na seção 2.11.

Este detetor requer somente 3(três) parâmetros. O primeiro é o σ , que é o desvio padrão do filtro Gaussiano especificado em *pixels*. Os outros dois parâmetros são: o limiar inferior e o limiar superior utilizados no processo de *non-maximal suppression*.

2.10.1.2 Rothwell

O algoritmo de Rothwell [35] é um método de detecção de bordas que utiliza um limiar dinâmico para cada *pixel* candidato a borda na imagem. Ele é similar ao algoritmo de Canny, pois aplica suavização pelo filtro Gaussiano seguido da diferenciação da direção da borda.

A diferença principal quando comparado ao algoritmo de Canny é que o algoritmo de Rothwell faz o processo de afinamento após a detecção de bordas. Ele utiliza o limiar dinâmico (imagem de limiares) para calcular os *pixels* de borda e depois faz o afinamento. O processo de afinamento é baseado no algoritmo de Tsao-Fu [41] que fornece um mapa de bordas com uma boa descrição topológica dos objetos.

A principal razão para não utilizar o processo de *non-maximal suppression* para o afinamento é que este processo não preserva as junções das bordas. Além disso, muitas vezes este processo fornece uma borda pontilhada ou serrilhada, que dificulta os processos seguintes, com por exemplo o fechamento.

Este algoritmo requer 3(três) parâmetros. Eles são o σ para a filtragem, um limiar inicial para as bordas, e um parâmetro α que adapta os *pixels* da imagem de limiares para incrementar a detecção dos *pixels* que estão perto de outras bordas para preservar as junções.

Além disso, Rothwell estende o trabalho de detecção de bordas e apresenta um método para recuperar informações de topologia a partir das bordas utilizando localização *subpixel* para obter uma melhor localização das bordas.

Recentemente, os trabalhos de Bellon et al. [4, 5] utilizaram uma adaptação do al-

goritmo de Rothwell para detecção de bordas e segmentação de imagens de profundidade. Nesta adaptação, observou-se que o algoritmo de Rothwell é muito sensível a ruídos.

2.10.1.3 Nalwa

O detetor de bordas criado por Nalwa e Binford [32] diferencia-se dos demais algoritmos estudados durante este trabalho, por utilizar ajuste de superfícies para detetar bordas.

Ele difere também da abordagem da filtragem linear usada no detetor de bordas Canny porque a derivada da imagem não é calculada. Ao invés disso, as funções da tangente hiperbólica (\tanh) e quadráticas são ajustadas na imagem de intensidade em uma janela 5x5 que é aplicado em toda a imagem.

Se o erro de ajuste da \tanh em um *pixel* $p(x, y)$ é menor que o erro de ajuste quadrático neste *pixel*, então o *pixel* é marcado como candidato a borda, caso contrário este *pixel* é marcado como “não borda”. Sobre os candidatos a *pixel* de borda é realizada uma filtragem, baseada em um limiar, para reduzir o número de *pixels* falsos, que não representam efetivamente uma borda.

O algoritmo de Nalwa gera uma saída que indica a direção e a localização *subpixel* de cada segmento de borda chamado *edgel*. Os *edgels* indicam a intensidade da borda e sua direção em um determinado *pixel* da imagem.

Durante este trabalho foram realizados testes e algumas alterações neste algoritmo de forma a poder trabalhar com um tamanho de janela genérico, não somente de tamanho 5x5. Também foram feitas modificações para obter uma imagem binária resultante com o mapa de bordas. Os testes mostraram que este algoritmo quando comparado a diversos algoritmos é sensivelmente mais robusto a ruídos.

2.10.2 Métodos de agrupamento

Métodos de agrupamento (*clustering*) têm como objetivo a classificação de amostras ou objetos em um espaço multidimensional de acordo com suas características intrínsecas, resultando em classes ou grupos bem definidos. Neste contexto, tais métodos vêm sendo freqüentemente utilizados para a segmentação de imagens, com o objetivo de agrupar

elementos da imagem que apresentam características similares.

Existem diversos métodos de agrupamentos com diferentes critérios e metodologias [10, 12, 22]. Os métodos de agrupamento muitas vezes demandam um tempo computacional grande, quando comparados a outros processos de segmentação.

A grande dificuldade dos algoritmos de agrupamento para a segmentação de imagens está em conseguir identificar superfícies distintas, porém com a mesma característica. Por exemplo, se considerarmos a característica orientação da superfície, duas regiões vizinhas que têm a mesma orientação podem ficar na mesma classe após o processo de agrupamento. A figura 2.10 mostra este problema nas regiões do centro da imagem, onde existe uma borda de orientação que pode guiar o processo de agrupamento e fornecer uma segmentação mais adequada [5].

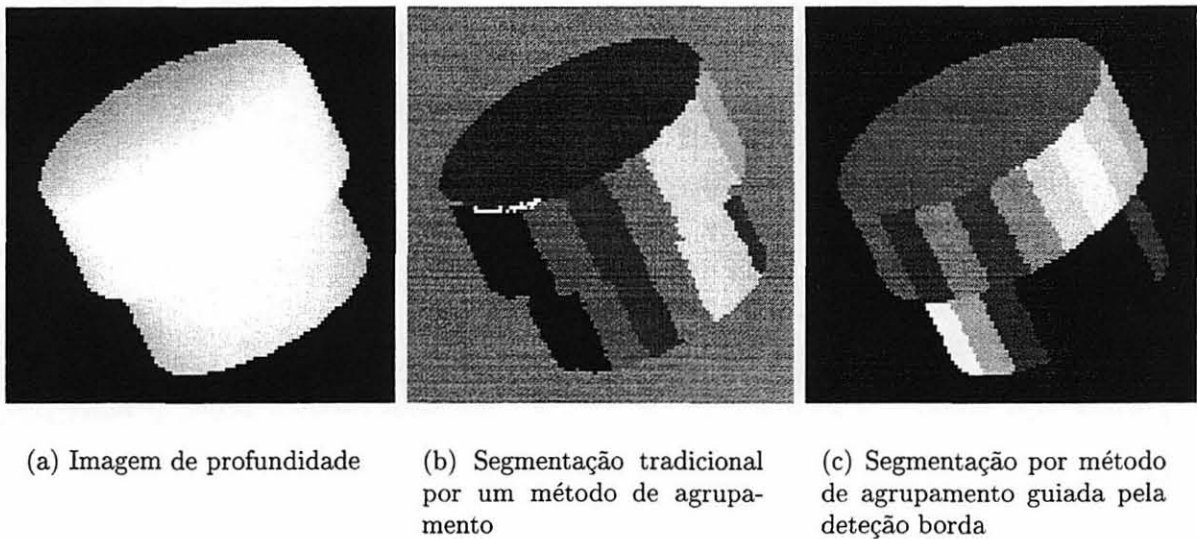


Figura 2.10: Exemplo de segmentação de imagens de profundidade.

O trabalho de Celinski [10], realizado recentemente, teve como objetivo descrever, classificar e comparar métodos de agrupamento para aplicação em segmentação de imagens. Um dos algoritmos de agrupamento mais usado é o algoritmo *K-means*.

K-means Este algoritmo tem a propriedade de agrupar os *pixels* da imagem em grupos (*clusters*) segundo a característica da imagem de entrada. Se a imagem de entrada for uma imagem de tons de cinza, então o algoritmo agrupa os *pixels* da imagem em n regiões que possuam valores de tons de cinza parecidos ou próximos. Cada região i possui um

rótulo, dado por um valor inteiro compreendido entre: $0 \leq i < n$.

No caso da segmentação de superfícies curvas, o algoritmo *K-means* pode utilizar o vetor normal de cada *pixel* da superfície como a característica para o processo de agrupamento. Desta maneira os *pixels* que tiverem orientações semelhantes são agrupados em uma mesma região. Neste caso, as superfícies são aproximadas por regiões planares.

Os parâmetros desse algoritmo são: o número de grupos (n) e o número máximo de iterações (T).

O algoritmo *K-means* pode ser definido pelos seguintes passos:

- Passo 1: Estabelece os valores dos n centros de grupo iniciais, arbitrariamente ou não;
- Passo 2: Distribui os elementos (*pixels*) entre os grupos atribuindo cada elemento ao grupo cujo centro possui característica mais próxima;
- Passo 3: Calcula centros de grupo dos grupos existentes, pela média de todos os componentes de cada grupo;
- Passo 4: Os passos 2 e 3 devem se repetir até que não haja mais trocas nos centros de grupos ou que se atinja o número máximo de iterações T definido.

Ao final deste processo, obtém-se a imagem segmentada em regiões, como pode ser visto na figura 2.10(b).

Algumas implementações deste algoritmo podem receber como parâmetro de entrada, também as posições dos centros de grupos iniciais, isso facilita a convergência do algoritmo otimizando o processo de agrupamento.

2.10.3 Crescimento de regiões

O crescimento de regiões (*region growing*) é o processo utilizado para agrupar *pixels* ou subregiões formando regiões significativas [17, 33].

A base do processo é a escolha de um *pixel* inicial, ou um pequeno conjunto de *pixels*, para a região e, a partir dele, englobar todos os *pixels* próximos que tenham propriedades

semelhantes às do *pixel* inicial. Este procedimento se repete novamente até todos os *pixels* da imagem serem agrupados em regiões.

O principal problema neste processo é a escolha do *pixel* inicial para cada região e o limiar para a agregação, que define se o *pixel* é ou não semelhante. Dependendo do valor de limiar para a agregação de *pixels* à região e a posição do *pixel* inicial, o resultado gerado pode ser diferente.

Para melhorar o desempenho dos algoritmos de crescimento de regiões, é necessário guiá-los de forma a manter a integridade das regiões dos objetos. Isso pode ser realizado agregando outra característica no processo, por exemplo a informação de bordas, como foi adotado neste trabalho e é apresentada no capítulo 4. Esta abordagem utiliza como característica de agregação apenas a conectividade de cada *pixel* com relação ao *pixel* inicial de cada região, não necessitando de limiares. Desta forma as regiões crescem até o limite de uma borda, evitando que regiões distintas sejam unidas erroneamente.

2.10.4 Divisão-e-união

Este método agrupa *pixels* em regiões sobre as quais um critério de homogeneidade C_h é verificado. Este critério pode ser, por exemplo, mesmo tom de cinza.

Inicialmente a imagem é dividida em 4 regiões e cada uma dessas regiões que não atenderem ao critério C_h é novamente divididas em outras quatro regiões e assim sucessivamente. Após a divisão faz-se a união das regiões adjacentes mais semelhantes. Nas regiões unidas onde o critério C_h não é verificado, o processo de divisão-e-união se repete até a sua convergência ou enquanto não ultrapassa um determinado número de repetições. Este processo fornece como resultado a imagem segmentada em regiões com a mesma característica [33].

2.11 Afinamento

Os algoritmos de afinamento são mais empregados em imagens binárias, por exemplo imagens de bordas.

Em alguns processos de detecção de borda o mapa de bordas resultante pode conter alguns *pixels* detetados erroneamente, fazendo com que a borda tenha uma espessura maior do que um *pixel*.

Para obter um mapa de bordas mais preciso, com bordas corretamente localizadas, que definam bem as regiões da imagem, é necessário afinar as bordas, de forma que fiquem com apenas um *pixel* de espessura.

Os algoritmos de afinamento mais utilizados são os baseados em métodos de esqueletização (*skeleton*), em Morfologia Matemática e no processo de *non-maximal suppression* [36, 28].

Alguns desses algoritmos são dependentes de limiar, e muitas vezes além de não preservarem a topologia, não geram uma borda com apenas um *pixel* de espessura.

Quando o processo de afinamento é aplicado baseado em Morfologia Matemática, a grande dificuldade é identificar os elementos estruturantes para que o algoritmo de afinamento tenha um resultado equivalente para diferentes tipos de imagens.

Uma abordagem empregada neste trabalho é utilizar um método híbrido baseado em elementos estruturantes da Morfologia Matemática e em métodos de esqueletização que é apresentada no capítulo 4.

Non-maximal suppression O processo de afinamento *non-maximal suppression* é utilizado em alguns detetores de bordas, como por exemplo, o detetor de bordas Canny, descrito na subseção 2.10.1.1.

A imagem de gradiente terá valores altos onde o gradiente da imagem for alto, mas isto não é suficiente para identificar corretamente a posição das bordas, haja visto que nem todos os valores de magnitude altos significam bordas. Para identificar as bordas, os valores de gradiente devem ser afinados de forma que somente os *pixels* de maior mudança local permaneçam.

O processo de *non-maximal suppression* fornece uma boa localização da borda por um processo de afinamento. Zero é colocado como o valor magnitude do gradiente de um *pixel* se a magnitude do gradiente deste *pixel* não é maior do que a magnitude de dois de

seus vizinhos na direção do gradiente. Ou seja, este processo elimina todos os valores ao longo da linha do gradiente que não são máximos.

Em seguida, para marcar os *pixels* de borda na imagem de bordas, utiliza-se dois limiares: um limiar inferior e um limiar superior. Todos os candidatos a *pixel* de borda que possuem magnitude do gradiente menor que o limiar baixo são marcados como “não bordas” e todos os *pixels* acima do limiar inferior e que podem ser vizinhos de algum *pixel* que está acima do limiar superior são marcados como bordas.

2.12 Fechamento

Quando observados os mapas de bordas gerados pelos detetores de bordas, nota-se que em algumas partes deste mapa existem bordas fragmentadas, principalmente nas junções das bordas que deveriam ter sido detetadas, mas não foram por algumas limitações durante o processo de identificação ou afinamento destas bordas.

Esse efeito é ocasionado, principalmente quando tipos diferentes de bordas são detetados separadamente e depois mesclados em um único mapa de bordas [17, 33].

Por exemplo em imagens de profundidade, quando combinadas as bordas de orientação e as bordas de profundidade em um único mapa de bordas, aparecem buracos ou fragmentos abertos de bordas nas junções. Isto ocorre devido ao fato de que estas junções não são detetadas nem como bordas de profundidade, nem como bordas de orientação, devido aos valores de limiar utilizados ou processos de suavização.

Portanto, para obter um mapa de bordas confiável é necessário um processo de fechamento deste mapa de bordas. A grande maioria dos algoritmos de fechamento são baseados em Morfologia Matemática, com operações sucessivas de dilatação e erosão de alguns elementos estruturantes.

Outra forma de executar o processo de fechamento é localizando os *pixels* $p_e(x, y)$ de extremidades da borda, ou seja, os *pixels* que possuem apenas 1(um) vizinho que também é um *pixel* de borda. Uma borda é traçada entre cada *pixel* $p_e(x, y)$ e um *pixel* de borda mais próximo que não seja adjacente ao próprio segmento de borda de $p_e(x, y)$. Este processo, conhecido como *edge link*, foi utilizado neste trabalho e os detalhes da sua implementação

são apresentados no capítulo 4.

CAPÍTULO 3

SEGMENTAÇÃO DE IMAGENS DE PROFUNDIDADE

Neste capítulo são apresentados alguns trabalhos relevantes relacionados à segmentação de imagens de profundidade.

Nos últimos anos, diversos grupos de pesquisa em processamento de imagens vêm desenvolvendo algoritmos para a extração de características de imagens de profundidade. Uma delas é a identificação das superfícies dos objetos da imagem, fornecida através da segmentação. Como a grande maioria dos pesquisadores utiliza um conjunto de imagens de diferentes tipos, tamanhos e objetos, o processo de avaliação destes algoritmos torna-se um desafio. Para facilitar esta avaliação, alguns trabalhos como o de Hoover et al. [21], estão voltados à definição de métricas utilizadas para comparar os diversos segmentadores quando submetidos a uma base de imagens específica. Desta forma, é possível avaliar diferentes metodologias para o problema da segmentação de imagens de profundidade.

A primeira seção deste capítulo aborda o problema da definição de segmentação quando aplicada a imagens de profundidade. Os trabalhos relacionados são apresentados na seção 3.2 e os métodos utilizados para comparação e avaliação de segmentadores de imagens de profundidade, na seção 3.3. Finalizando com uma conclusão, na seção 3.4.

3.1 Definição do problema

Genericamente, a segmentação de imagens de profundidade é o processo de rotular todos os *pixels* que estão em uma mesma região com o mesmo rótulo, de acordo com o predicado de homogeneidade definido.

O grande problema está em determinar qual a característica destes *pixels* será considerada para o processo de rotulação, de forma que a segmentação identifique corretamente superfícies distintas da imagem.

Se a definição de segmentação de imagens de profundidade for a mesma da definição de

segmentação apresentada na seção 2.10, algumas inconsistências aparecem. Por exemplo, nas imagens de profundidade reais alguns *pixels* não contém medidas de profundidade precisas [21]. Estes *pixels* devem ser considerados como uma única região de ruídos, porém esta região pode não ser conexa, resultando diversas pequenas regiões distintas na imagem segmentada, que possuem a mesma característica.

Os algoritmos apresentados na seção seguinte, foram submetidos à avaliação para a segmentação de diversas imagens de diferentes tipos (ABW e Perceptron).

3.2 Trabalhos relacionados

Nesta seção, são apresentados 4(quatro) algoritmos de segmentação de imagens de profundidade recentemente testados, avaliados e publicados. Os algoritmos foram escolhidos porque possuem diferenças significativas na sua metodologia. Estes algoritmos foram submetidos apenas à avaliação da segmentação de imagens de profundidade compostas por objetos poliédricos. O método de avaliação será discutido na seção 3.3.

Os algoritmos estão identificados com as iniciais dos nomes das instituições de pesquisa dos autores:

- 1) (USF) University of South Florida, USA.
- 2) (WSU) Washington State University, USA.
- 3) (UB) University of Bern, Switzerland.
- 4) (UE) University of Edinburgh, Scotland.

3.2.1 USF

O segmentador USF [21] pode ser caracterizado como um algoritmo de segmentação de imagens baseado no crescimento de regiões. Basicamente, no primeiro estágio é calculado um ajuste planar para cada *pixel* da imagem e faz-se o crescimento de regiões unindo os *pixels* que possuem equações de planos similares.

Um segundo estágio é utilizado para calcular os coeficientes do vetor normal à superfície de cada *pixel* da imagem. Para cada *pixel* $p(x, y)$ da imagem, são separados

os conjuntos dos *pixels* válidos (*inliers*) $P_{inliers}[p(x, y)]$ dos *pixels* inválidos (*outliers*) $P_{outliers}[p(x, y)]$ em uma janela de tamanho $N \times N$ centrada em $p(x, y)$. Os *pixels* válidos são aqueles que têm a diferença do seu valor de profundidade com o valor de profundidade de $p(x, y)$ inferior a um limiar T_{perp} .

Para identificar os *pixels* válidos $P_{inliers}[p(x, y)]$, uma operação de crescimento é feita a partir dos vizinhos de $p(x, y)$ que está no centro da janela, marcando todos os *pixels* dentro da janela $N \times N$ que estão abaixo de T_{perp} . Isto é necessário para eliminar dos cálculos os *pixels* que estão entre uma borda de profundidade e os *pixels* que são ruídos.

Se menos que 50% dos *pixels* da janela são válidos, então uma única equação de plano é ajustada a estes *pixels*. Se 50% ou mais dos *pixels* são válidos, então um conjunto de 9(nove) equações do plano são calculadas usando os *pixels* válidos $P_{inliers}[p(x, y)]$. As nove equações são calculadas nas direções horizontais, verticais, diagonais e central da janela. A equação do plano que tiver o menor erro residual é a equação que fornecerá os coeficientes do vetor normal para o *pixel* $p(x, y)$. O uso das nove equações produz geralmente vetores normais mais precisos para os *pixels* próximos a bordas de orientação, do que o vetor produzido por apenas uma equação do plano.

Após calculados os valores dos vetores normais mais adequados, o *pixel* que tiver o menor erro residual é considerado como o *pixel* inicial para o processo de crescimento de regiões. A equação do plano desta região é calculada inicialmente pelo vetor normal deste *pixel* inicial e a sua localização espacial de coordenadas (x, y, z) . Os critérios para a união dos *pixels* no processo de crescimento são:

- 1) Ser de uma vizinhança 4 a um *pixel* da região crescida.
- 2) O ângulo entre a normal do *pixel* e a normal da região deve estar abaixo de um limiar T_{angle} .
- 3) A distância perpendicular entre o *pixel* e a equação do plano da região deve estar abaixo de um limiar T_{perp} .
- 4) A distância ponto a ponto entre o *pixel* e seu vizinho (vizinhança 4) que já está conectado a região deve estar abaixo de um limiar T_{point} .

O crescimento é realizado até que não existam mais *pixels* a serem unidos à primeira região. Este processo se repete (um novo *pixel* inicial é gerado para uma nova região) até todos os *pixels* da imagem estarem associados a uma região.

Uma vez que todas as regiões foram delimitadas, um refinamento final é aplicado para obter a orientação de cada uma das regiões. Para cada região cujo tamanho alcançar $\frac{N^2}{4}$, uma equação do plano é calculada usando todos os *pixels* pertencentes à região. Se a região não alcançar este tamanho mínimo, e seu tamanho estiver abaixo de um limiar T_{area} , então ela é descartada e seus *pixels* não são mais considerados.

Este algoritmo tem como ponto positivo o cálculo eficiente dos valores dos vetores normais utilizando as nove equações do plano, principalmente em regiões de bordas de orientação. Entretanto, o processo de crescimento de regiões é custoso pelo número de critérios utilizados para a união dos *pixels* e pouco eficiente originando um grande número de pequenas regiões. O resultado final da segmentação apresenta uma boa descrição topológica das superfícies dos objetos, porém com uma grande quantidade de regiões de ruído, que poderia ser minimizada com um processo de filtragem inicial da imagem.

3.2.2 WSU

O algoritmo WSU [14] é baseado no trabalho Hoffman e Jain [20], mas contém alguns melhoramentos incorporados por Flynn [14]. Este algoritmo não é otimizado para objetos poliédricos, porque faz ajuste de superfícies quádricas. Foram feitas algumas modificações para poder trabalhar especificamente com superfícies planares.

O segmentador WSU é formado pelas seguintes fases:

- 1) Identifica os *pixels* de borda de profundidade através de um limiar t_j . Se a diferença máxima de profundidade entre o *pixel* $p(x, y)$ e qualquer um de seus 8 vizinhos for maior que t_j então $p(x, y)$ é marcado como *pixel* de borda.
- 2) Calcula os vetores normais às superfícies para cada *pixel* $p(x, y)$ que não tenha um *pixel* de borda dentro da janela $N \times N$ centrada em $p(x, y)$. Estes vetores normais são calculados através do ajuste de uma componente principal para todos os *pixels* da

janela [15]. A direção principal com a menor variância é considerada o vetor normal à superfície. Esta técnica é muito eficaz para dados contaminados por ruídos nas 3(três) coordenadas (x,y,z).

- 3) Agrupa os dados à posição de cada *pixel* da imagem e os coeficientes do vetor normal em uma matriz de 6(seis) dimensões, que é utilizada como entrada para um algoritmo de agrupamento, chamado CLUSTER [10]. O algoritmo CLUSTER, de forma similar ao *K-means*, basicamente agrupa em regiões os *pixels* que tenham uma orientação similar, baseado no erro quadrático.
- 4) Converte regiões agrupadas em uma imagem segmentada, rotulando cada *pixel* com o valor do centro de grupo mais próximo. Componentes conectados são então encontrados para evitar rótulos idênticos para regiões distintas na imagem. A imagem resultante é sobre-segmentada (*over-segmentation*).
- 5) Executa um processo de união (*merging*) de duas regiões segmentadas adjacentes em uma mesma região. Este processo é realizado para todos os segmentos adjacentes, cuja média angular entre os vetores normais dos *pixels* da interseção dos segmentos é menor que um limiar t_θ . Quando a imagem de profundidade possui apenas objetos poliédricos, a segmentação neste ponto está muito próxima da segmentação final.
- 6) Classifica cada segmento como planar ou não planar. O mesmo ajuste de uma componente principal utilizado no passo 2 é aplicado para todos os *pixels* de um segmento, e o erro quadrático mínimo é calculado para este ajuste. Se o erro for maior que um limiar t_p , este segmento é classificado como não planar e ignorado em processos seguintes.
- 7) Aplica novamente um processo de união para unir segmentos que são adjacentes e têm parâmetros similares. Especificamente, segmentos planares são unidos se a diferença angular entre suas orientações é menor que t_a e a distância dos termos das suas equações implícitas diferem menos que um limiar t_d .
- 8) Realiza a união dos *pixels* na fronteira de cada segmento, que não foram rotulados,

a este mesmo segmento se o erro de ajuste destes *pixels* é menor que t_f . Este processo faz com que *pixels* rotulados inicialmente como não planares, vizinhos a uma superfície planar possam ser marcados como planares.

9) Repete os três passos acima até não haver mais mudanças de rótulos.

Neste algoritmo pequenas regiões de ruídos podem ser criadas após o processo de agrupamento. Para remover estas regiões, um simples processo que analisa a conectividade de cada *pixel* e remove todas as regiões cujo tamanho seja menor que um limiar t_s pode ser usado para refinamento da segmentação.

O grande problema deste algoritmo é o tempo de execução, pois é dramaticamente lento. Quando foi implementado, os autores utilizaram imagens bem menores (até 4 vezes menores) do que as imagens ABW e não tiveram a preocupação com o tempo de execução do algoritmo que não era tão ruim para as suas imagens teste. Este tempo de processamento se agrava também pelo grande número de processos intermediários de crescimento e união para o refinamento das regiões.

Este algoritmo possui muitos limiares, o que dificulta o ajuste desses valores para a obtenção de resultados equivalentes para tipos diferentes de imagens. Na segmentação final existe uma quantidade razoável de superfícies com ruído, mas uma boa descrição topológica das superfícies dos objetos.

3.2.3 UB

O segmentador UB [25] é baseado no fato de que, num caso ideal, os pontos em uma linha de varredura (*scan line*) de uma imagem de profundidade pertencentes a uma superfície planar, formam um segmento linear reto 3D. Em outras palavras, todos os pontos no segmento linear 3D seguramente pertencem a uma mesma superfície planar. O processo basicamente divide cada linha de varredura em segmentos lineares, e faz um crescimento de regiões agrupando os segmentos lineares ao invés de *pixels* individuais.

O primeiro passo do algoritmo utiliza um simples método de divisão (*split*) que recursivamente divide cada linha de varredura em segmentos lineares de forma que a distância

perpendicular entre os pontos da linha de varredura e os seus correspondentes no segmento linear estejam abaixo de um limiar T_s eg.

Após esta divisão, é realizado um processo de crescimento de regiões. O ponto inicial para o crescimento é um conjunto formado por segmentos lineares de 3(três) linhas de varredura vizinhas que satisfaz 3(três) condições:

- 1) Todos os três segmentos lineares devem ter um tamanho maior que um limiar t_1 .
- 2) A parte sobreposta de dois segmentos lineares vizinhos deve ter um tamanho maior que um limiar t_2 que é definido como uma percentagem do tamanho de cada segmento linear.
- 3) A distância de cada par de pontos vizinhos, cada um pertencente a um segmento linear distinto, deve ser menor que o limiar t_3 .

O candidato que tiver a maior soma dos tamanhos dos segmentos é escolhido como o ponto inicial para o crescimento de regiões.

No processo de crescimento, um segmento linear é adicionado a uma região se a distância perpendicular entre seus dois pontos de extremidade e a equação do plano da região não for maior que um limiar definido por:

$$T_2 + \frac{t_4 \times \text{size}}{10000} \quad (3.1)$$

Onde:

size é o número de *pixels* da região expandida no momento.

T_2 e t_4 são limiares para facilitar a expansão para regiões muito grandes.

Quando não existirem segmentos a serem adicionados a esta região, um novo ponto inicial é calculado para começar outro processo de crescimento, formando uma nova região. Este procedimento se repete até todos os segmentos serem distribuídos em regiões.

As regiões que tiverem tamanho menor que um limiar t_5 são descartadas neste processo final de refinamento.

Este algoritmo é o mais eficiente entre os demais com relação ao tempo de execução. Entretanto, as bordas entre as regiões segmentadas não são bem definidas, não preservando a topologia das superfícies. Apesar de ser robusto a ruídos, em regiões pequenas a segmentação é pouco eficiente com relação às junções entre estas superfícies. Existe ainda uma dependência grande de limiares entre os processos.

3.2.4 UE

Este segmentador UE [2, 1] é baseado no processo de crescimento de regiões e é semelhante ao segmentador USF.

Ele pode ser descrito através de 4(quatro) estágios básicos:

- 1) Calcula os vetores normais para cada *pixel* usando um ajuste de plano para todos os *pixels* de uma janela de tamanho 5x5. As discontinuidades de profundidade e de orientação são calculadas usando dois limiares T_d e T_o respectivamente. O limiar T_d é a diferença mínima de profundidade entre o valor de profundidade de dois *pixels* e o limiar T_o é a diferença angular mínima entre os valores dos vetores normais de dois *pixels*. Após este processo, uma suavização é feita na imagem com uma filtragem que preserva as discontinuidades (filtro mediano).
- 2) Aplica um processo de segmentação baseado no cálculo da curvatura K e H nos *pixels* de uma janela NxN. A curvatura K é a curvatura Gaussiana e a curvatura H é a curvatura média [7]. Os *pixels* são marcados segundo seu tipo de superfície (elíptica, planar, etc) baseado na combinação dos sinais dos valores das curvaturas K e H . Cada valor de curvatura é classificado como: Negativo, Zero, Positivo ou Desconhecido baseado em dois limiares T_{inner} e T_{outer} . O limiar T_{inner} determina o limite máximo de valores classificados como Zero e o limiar T_{outer} determina o limite mínimo de valores Positivos e Negativos. Entre estes valores o *pixel* é classificado como Desconhecido. Uma vez que os *pixels* são rotulados com os sinais de K e H , os *pixels* de uma vizinhança 8 são agrupados para formar as regiões iniciais. Nesta segmentação inicial são então aplicadas as operações de dilatação e

erosão da Morfologia Matemática, de forma a preencher pequenas áreas classificadas como Desconhecidas, eliminar pequenas regiões e separar componentes fracamente conectados.

- 3) Calcula um ajuste de superfície, baseado nos mínimos quadrados, para cada região da segmentação inicial cujo tamanho está acima de um limiar T_s . Um crescimento de regiões é realizado em três etapas sucessivas: expansão, refinamento e contração. Para a expansão, um *pixel* é adicionado à região se preenche os seguintes critérios:

- Tem um *pixel* da região na sua vizinhança 8.
- O *pixel* tem uma distância perpendicular da superfície menor que um limiar t_{min} .
- O *pixel* é mais próximo à superfície da região do que da superfície inicial.
- A diferença angular entre o vetor normal do *pixel* e o vetor normal da superfície naquele ponto deve ser menor do que um limiar t_n .
- O diferença angular entre o vetor normal do *pixel* e a normal da nova região deve ser menor do que a diferença do seu vetor normal e a normal de sua região inicial.

Feita esta expansão, a superfície desta região é novamente ajustada, nesta etapa de refinamento. Finalmente, é aplicado o processo de contração da região, utilizando para cada *pixel* da região o mesmo critério da expansão em relação a nova superfície ajustada. Se o *pixel* não preenche os critérios, então ele volta a ser rotulado com a sua região inicial atribuída antes do processo de expansão. Estas três etapas se repetem até que não haja mais modificações na rotulação dos *pixels* ou um número máximo de iterações seja alcançado, definido por um limiar T_{max} .

- 4) O último estágio é o refinamento das bordas das regiões, necessário para rotular eventuais *pixels* da interseção de duas regiões que estão sem rótulo. Isto é realizado usando a mesma abordagem do estágio anterior (expansão, refinamento e contração

sucessivas), porém com diferentes critérios para a inclusão dos *pixels*. Neste caso, um *pixel* é adicionado a uma região durante o processo de expansão se:

- Tem um *pixel* da região na sua vizinhança 8.
- O *pixel* tem uma distância à superfície da região menor que um limiar t_r .

Como no crescimento de regiões do estágio anterior, o processo de refinamento e contração seguem os mesmos critérios. Este último estágio de refinamento das bordas é aplicado uma única vez, entretanto se necessário for, pode ser aplicado outras vezes de forma a melhorar a segmentação final.

Apesar das imagens testes ABW possuírem apenas objetos poliédricos, este algoritmo também pode ser utilizado para a segmentação de superfícies curvas. Isto é realizado baseado no cálculo das curvaturas K e H que classificam diferentes tipos de superfícies.

Este algoritmo também tem a desvantagem de possuir muitos limiares, o que dificulta a obtenção de resultados equivalentes com diferentes tipos de imagens. Entretanto, o algoritmo mostra bons resultados quanto à topologia apesar de não preservar pequenas regiões significativas da imagem, devido aos vários processos sucessivos de dilatação e erosão da Morfologia Matemática. O seu tempo de execução é em geral lento, podendo ser mais eficiente dependendo do limiar para o número de iterações do processo de crescimento e refinamento.

3.3 Métodos de avaliação

Existem diversas características que podem ser consideradas no processo de comparação de resultados da segmentação de imagens. Muitos trabalhos destacam algumas destas características no resultado da segmentação, mas poucos realmente definem métricas para uma comparação destas características comuns entre os segmentadores. Isto é mais evidenciado nos trabalhos que utilizam imagens de profundidade.

Uma dificuldade adicional neste processo está na grande variedade de sensores e tipos de imagens existentes e utilizados pelos grupos de pesquisa de segmentação de imagens

de profundidade. A identificação e utilização de um grupo de imagens, para submeter os algoritmos à avaliação, é de grande importância.

No primeiro trabalho específico sobre comparação de segmentadores de imagens de profundidade [21], a avaliação foi feita apenas para imagens com superfícies planares. Como não havia sido apresentado nenhum comparador de segmentadores de imagens de profundidade, os autores limitaram o trabalho a superfícies planares considerando-se que nem esse problema, aparentemente mais simples, tem solução fechada. Com o resultado comparativo desde trabalho inicial de comparação, outros trabalhos abrangendo superfícies curvas poderiam ser realizados posteriormente.

Alguns grupos de pesquisa estão concentrando esforços para a criação de métodos de avaliação e algoritmos de segmentação de imagens de profundidade, também aplicados a imagens com superfícies curvas.

No trabalho de Hoover et al. [21] foram definidos os tipos de imagens a serem processadas, o conjunto de métricas para a avaliação e uma ferramenta para geração, formatação e comparação automática dos resultados da avaliação dos segmentadores. Foram definidas como as imagens de avaliação: 40 imagens do tipo ABW e 40 imagens do tipo Perceptron. Cada imagem contém no máximo 5(cinco) objetos poliédricos posicionados em diferentes posições e com diferenças angulares entre os objetos. As imagens foram aleatoriamente divididas em dois grupos: 10 imagens de treinamento e 30 imagens de teste, para cada tipo de imagem. As imagens de treinamento são para o ajuste de limiares dos algoritmos antes de serem submetidos às imagens teste, cujos resultados foram avaliados.

Para a avaliação dos resultados gerados pelos algoritmos, para cada uma das 80 imagens, foram criadas imagens segmentadas manualmente (*ground truth segmentation*), por uma pessoa. Estas imagens consistem basicamente da segmentação manual e um conjunto de ângulos.

No processo de rotulação das regiões segmentadas manualmente, foram reservados 10 valores de rótulos para vários tipos de *pixels* não considerados como sendo de superfícies (*non-surface pixel*), mas somente 4 foram definidos até o momento:

- sombra (*shadow*): para os *pixels* onde o sensor não conseguiu obter a informação de

profundidade, normalmente ocorre nas imagens ABW.

- ruído (*noise*): para os *pixels* onde o sensor fez uma medida errada.
- borda cruzada (*cross-edge*): para os *pixels* onde o sensor captura a informação de profundidade de mais do que uma superfície.
- indefinido (*undiscernable surface detail*): para os *pixels* onde o sensor faz a leitura de valores de profundidade, mas não tem informação suficiente para encontrar a separação das superfícies. Pode ocorrer por uma quantização insuficiente.

Os *pixels* não considerados como sendo de superfícies não representam regiões, e não são avaliados neste método.

Cada imagem segmentada manualmente foi revisada por uma segunda pessoa, para eliminar eventuais erros. Finalmente, o ângulo entre as regiões segmentadas foi calculado.

A avaliação entre a imagem de profundidade segmentada pelo algoritmo (*MS*) e a imagem segmentada manualmente (*GT*) é feita da seguinte forma:

Seja M o número de regiões em *MS* e N o número de regiões em *GT*. N não inclui os *pixels* não considerados superfícies. Similarmente, M não inclui nenhum *pixel* não rotulado pelo segmentador.

Seja P_m o número de *pixels* em cada região R_m segmentada pelo algoritmo, onde $m = \{1, \dots, M\}$. Similarmente, seja P_n o número de *pixels* em cada região R_n segmentada manualmente, onde $n = \{1, \dots, N\}$.

Seja $O_{mn} = R_m \cap R_n$ o número de *pixels* que possuem as mesmas coordenadas em ambas as regiões R_m e R_n .

Então, se não existe sobreposição entre duas regiões, $O_{mn} = 0$, e se existe uma sobreposição total, $O_{mn} = P_m = P_n$.

Uma tabela $M \times N$ é criada, contendo O_{mn} para $m = \{1, \dots, M\}$ e $n = \{1, \dots, N\}$. Implicitamente são anexadas para cada entrada na tabela a percentagem de sobreposição em relação ao tamanho de cada região. $\frac{O_{mn}}{P_m}$ representa a percentagem de m na interseção coberta por m e n . Similarmente, $\frac{O_{mn}}{P_n}$ representa a percentagem de n na interseção

coberta por m e n . Estas percentagens são usadas para determinar a classificação das regiões segmentadas.

São considerados 5(cinco) tipos de classificação para as regiões:

- deteção correta (*correct detection*): é usada para as superfícies corretamente segmentadas pelo segmentador.
- sobre-segmentação (*over-segmentation*): é usada no caso de múltiplas deteções de uma única superfície que resultam em uma topologia incorreta.
- sub-segmentação (*under-segmentation*): é usada no caso de uma separação insuficiente de múltiplas superfícies que resulta em um subconjunto de topologia correta e geometria deformada.
- perdida (*missed*): é usada quando o segmentador não encontra uma superfície que aparece na imagem, podem ser chamadas de falsas negativas.
- ruído (*noise*): é usada quando o segmentador supõe a existência de uma superfície que não está na imagem, podem ser chamadas de falsas positivas.

É importante notar que estas métricas podem ter mais ou menos relevância para diferentes aplicações, por exemplo, na descrição topológica das superfícies, o ruído pode ter menor relevância do que as superfícies perdidas.

As fórmulas para a decisão da classificação são baseadas em uma tolerância dada por um limiar T , onde $0.5 < T \leq 1.0$. O valor de T é a percentagem de aceitação de uma determinada classificação, que define a qualidade dos resultados.

As métricas definidas para cada classificação são:

1. Classificação - deteção correta.

Um par de regiões R_n na imagem GT e R_m na imagem MS são classificadas como deteção correta se:

- $O_{mn} \geq T \times P_m$ (pelo menos $T\%$ dos *pixels* na região R_m da imagem MS são marcados como *pixels* na região R_n da imagem GT), e

- $O_{mn} \geq T \times P_n$.

2. Classificação - sobre-segmentação.

A região R_n na imagem GT e um conjunto de regiões na imagem MS , dado por $\{R_{m_1}, R_{m_2}, \dots, R_{m_x}\}$, onde $2 \leq x \leq M$, são classificados como sobre-segmentação se:

- $\forall i \in x, O_{m_i n} \geq T \times P_{m_i}$ (pelo menos $T\%$ dos *pixels* em cada região R_{m_i} na imagem MS são marcados como *pixels* na região R_n na imagem GT), e
- $\sum_{i=1}^x O_{m_i n} \geq TP_n$ (pelo menos $T\%$ dos *pixels* na região R_n da imagem GT são marcadas como *pixels* na união das regiões $\{R_{m_1}, R_{m_2}, \dots, R_{m_x}\}$ na imagem MS).

3. Classificação - sub-segmentação.

Um conjunto de regiões na imagem GT dado por $\{R_{n_1}, R_{n_2}, \dots, R_{n_x}\}$, onde $2 \leq x \leq M$ e uma região R_m da imagem MS são classificados como sub-segmentação:

- $\sum_{i=1}^x O_{m_i n} \geq TP_m$ (pelo menos $T\%$ dos *pixels* na região R_m da imagem MS são marcadas como *pixels* na união das regiões $\{R_{n_1}, R_{n_2}, \dots, R_{n_x}\}$ na imagem GT), e
- $\forall i \in x, O_{m_i n} \geq T \times P_{n_i}$ (pelo menos $T\%$ dos *pixels* em cada região R_{n_i} na imagem GT são marcados como *pixels* na região R_m na imagem MS).

4. Classificação - perdida.

Uma região R_n na imagem GT que não é participante de nenhuma classificação - detecção correta, sobre-segmentação ou sub-segmentação é classificada como perdida.

5. Classificação - ruído.

Uma região R_m na imagem MS que não é participante de nenhuma classificação - detecção correta, sobre-segmentação ou sub-segmentação é classificada como ruído.

Uma vez que todas as regiões estejam classificadas, uma métrica final descreve a precisão da geometria recuperada e é calculada como:

Cada par de regiões R_{n_1} e R_{n_2} na imagem GT que representam faces adjacentes de um mesmo objeto na cena, tem a diferença angular A_n previamente calculada. Se R_{n_1} e R_{n_2} estão ambos classificados como detecção correta, então o ângulo A_m entre o vetor normal à superfície de suas regiões correspondentes na imagem MS é calculado. O valor absoluto da diferença entre estes dois ângulos é calculado como: $|A_n - A_m|$. Este cálculo é realizado para todas as regiões classificadas como detecção correta. O número de comparações de ângulos, o erro médio e o desvio padrão são armazenados para a segmentação de cada segmentador. Estas medidas fornecem uma estimativa indireta da precisão da geometria recuperada das regiões da imagem corretamente segmentadas.

A ferramenta criada para a comparação automática dos resultados, compara a imagem manualmente segmentada e da imagem segmentada gerada pelo segmentador, utilizando as métricas descritas anteriormente. Esta ferramenta foi utilizada para gerar todos os resultados apresentados no trabalho de Hoover et al. [21] e alguns deste trabalho que são apresentados no capítulo 5.

3.4 Conclusão

A comparação feita no trabalho de Hoover, tem uma importante contribuição que é o levantamento do estado da arte de segmentadores de imagens de profundidade. E baseado nos resultados, é possível notar claramente que o problema da segmentação planar de imagens de profundidade não está resolvido, e tão pouco para a segmentação de superfícies curvas.

É necessário ainda definir a importância das medidas de precisão da geometria de cada objeto, de maneira a guiar alguns processos futuros como a representação e o reconhecimento de objetos.

Com este estudo inicial dos diversos métodos de segmentação de imagens de profundidade, é possível levantar as principais vantagens e limitações de cada um dos algoritmos apresentados. Desta forma, este conjunto de informações ajuda no desenvolvimento de novas técnicas que forneçam resultados melhores e muitas vezes mais rápidos.

Com relação aos limiares dos algoritmos, praticamente todos são limiares fixos, o que

difículta a criação de algoritmos independentes das imagens de entrada. Como apresentado na seção 2.9 do capítulo 2, algumas aplicações necessitam de limiares adaptativos que sejam calculados automaticamente, gerando assim uma aplicação que melhor se adapta em diferentes ambientes. Neste caso, os algoritmos muitas vezes são mais complexos, demandando um tempo de processamento maior, mas que muitas vezes pode ser aceitável.

Outro problema, semelhante aos limiares, são os tamanhos das máscaras utilizados e definidos para cada um dos algoritmos. As máscaras de tamanho pequeno fornecem melhores resultados com relação à preservação da topologia das superfícies dos objetos, principalmente nas regiões de bordas, mas são mais sensíveis a ruídos. Outro problema é que algumas características da imagem podem não ser identificadas com máscaras pequenas. Por exemplo, na interseção de duas superfícies separadas por uma pequena diferença angular de orientação. Nesta região de interseção existe uma borda de orientação suave que é quase que imperceptível se for utilizada uma máscara de tamanho pequeno. Se o tamanho da máscara aumenta, uma quantidade maior de pontos é analisada facilitando a identificação desta interseção.

Quando são usadas máscaras maiores, existe uma imunidade maior a ruídos, entretanto a informação topológica pode ser prejudicada em algumas regiões da imagem. Portanto, existe também a necessidade de utilizar máscaras com tamanhos ajustáveis, calculados automaticamente para situações diferentes. O problema é saber onde utilizar um tamanho diferente de forma rápida, sem prejudicar a aplicação.

Geralmente os algoritmos de agrupamento são processos lentos e muito dependentes de limiares. Já os processos de crescimento de regiões são mais rápidos, mas também dependem de limiares em seus critérios de agregação.

CAPÍTULO 4

SEGMENTAÇÃO DE IMAGENS DE PROFUNDIDADE POR DETEÇÃO DE BORDAS

Neste capítulo são vistos os objetivos do algoritmo de segmentação de imagens de profundidade por detecção de bordas desenvolvido, a descrição de todos os métodos utilizados, a avaliação deste algoritmo e uma breve conclusão.

Neste trabalho foi utilizada a base de dados de imagens ABW do projeto de comparação de segmentação de imagens de profundidade ¹. As imagens desta base são compostas por diferentes tipos de imagens reais de objetos com superfícies planares.

A escolha pelo uso destas imagens deve-se, principalmente, ao fato de que muitos grupos de pesquisa em visão computacional e processamento de imagens vêm utilizando-as e fornecendo resultados para comparação de algoritmos de segmentação de imagens de profundidade [21]. As imagens do tipo Perceptron, por terem um nível de ruído muito elevado no processo de aquisição, não foram consideradas nos testes. O trabalho de Hoover et al. [21] mostrou que as imagens Perceptron não fornecem uma boa descrição das superfícies dos objetos na imagem de profundidade, sendo inviáveis para diversas aplicações.

Os resultados experimentais deste algoritmo aplicado às imagens de profundidade ABW são apresentados no capítulo 5, assim como a comparação com outros algoritmos apresentados no capítulo 3.

4.1 Objetivos

Com o estudo dos diversos métodos utilizados para a segmentação de imagens de profundidade, é possível levantar uma série de características importantes que devem ser consideradas em um segmentador para estas imagens.

¹<http://marathon.csee.usf.edu/range/seg-comp/SegComp.html>

Os objetivos principais do método desenvolvido são:

1. Resolver alguns dos problemas de segmentação de imagens de profundidade levantados na seção 3.4 do capítulo 3, de forma que o método se torne mais genérico e com pouca dependência de limiares fixos. Sem esta dependência, o algoritmo pode ser aplicado a um conjunto maior de imagens.
2. Fornecer resultados que preservem a topologia das superfícies dos objetos.
3. Ser robusto a ruídos.
4. Gerar uma segmentação com apenas as regiões significativas da imagem, sem regiões de ruídos e de maneira eficiente.

Com estes objetivos alcançados é possível fornecer uma segmentação de qualidade que facilitará os processos de representação e reconhecimento posteriores.

4.2 Método desenvolvido

Neste trabalho foi desenvolvido um novo método para a segmentação de imagens de profundidade baseado em técnicas de detecção de bordas. Além de diversas comparações e avaliações do método, a partir de sua aplicação a imagens compostas apenas por superfícies planares, também foram obtidos resultados para imagens com objetos não poliédricos.

O algoritmo possui 4 etapas principais para a segmentação de imagens com objetos poliédricos: a filtragem, detecção de bordas, refinamento do mapa de bordas e o crescimento de regiões. Uma quinta etapa pode ser considerada para a segmentação de imagens com superfícies curvas: por exemplo, aproximação por planos das superfícies curvas por algoritmos de agrupamentos ou aproximação por quádras. Cada uma das 4 etapas é apresentada e discutida nas subseções seguintes.

4.2.1 Filtragem

A filtragem é uma etapa muito importante neste método. Em imagens reais os ruídos dificultam a identificação com precisão de algumas características da imagem, como por

exemplo as bordas. Nas regiões de ruído, muitos *pixels* que não são verdadeiros *pixels* de borda podem ser marcados como tal, dificultando assim a preservação da topologia dos objetos.

O principal objetivo desta etapa é eliminar os ruídos impulsivos da imagem. O problema é identificar qual o filtro mais adequado, o tamanho de máscara necessário e quantas vezes este filtro será aplicado na imagem. Alguns testes realizados e comprovados em outros trabalhos, mostraram que um filtro mediano com uma máscara pequena pode suavizar satisfatoriamente ruídos impulsivos. Nas imagens de profundidade ABW, pode-se identificar este tipo de ruído [17, 33].

Então nesta primeira etapa foi utilizado um filtro mediano de tamanho 3×3 , o qual é aplicado uma única vez na imagem, nas 3(três) bandas (X, Y, Z). Uma outra abordagem pode ser utilizada para suavizar apenas os *pixels* onde realmente os ruídos são identificados, não suavizando a imagem toda. Este processo é aplicado para cada *pixel* que tenha valor menor ou maior que todos os seus *pixels* vizinhos. Somente nestes *pixels* é que o processo de suavização pelo filtro mediano é realizado de maneira a preservar a informação original da imagem obtida pelo sensor. Testes comprovaram que para as imagens ABW, o filtro mediano aplicado não causou perdas significativas de informação de maneira que esta outra abordagem não precisou ser utilizada nos experimentos.

4.2.2 Detecção de bordas

Nesta etapa são identificados 2(dois) tipos de bordas, as bordas de profundidade e as bordas de orientação, mostradas na figura 2.9 da seção 2.8.

Primeiramente são identificadas as bordas de profundidade e depois as bordas de orientação. Estes dois processos independentes são apresentados a seguir nas próximas subseções.

4.2.2.1 Bordas de profundidade

Neste processo apenas a banda (Z) da imagem de profundidade é utilizada. Considere que a $I_Z(x, y)$ é a imagem da banda (Z) da imagem de profundidade $I_{XYZ}(x, y)$ de tamanho

MxN.

Este processo é realizado da seguinte forma:

1. Calcula-se o gradiente de $I_Z(x, y)$, gerando uma imagem de gradientes $M_g(x, y)$, similar à fórmula 2.10. O valor do gradiente é maior nas regiões de bordas de profundidade.
2. Calcula-se um limiar automático T_{step} baseado no trabalho de Haddon [18]. Este limiar é calculado sobre a imagem $M_g(x, y)$ como:

$$T_{step} = \text{média} + \text{desvio}_p \cdot C_{step} \quad (4.1)$$

Onde:

$$\begin{aligned} \text{média} &= \frac{\sum_{x=0}^M \sum_{y=0}^N M_g(x, y)}{M \times N} \\ \text{desvio}_p &= \sqrt{\frac{\sum_{x=0}^M \sum_{y=0}^N (M_g(x, y) - \text{média})^2}{(M \times N) - 1}} \end{aligned} \quad (4.2)$$

E C_{step} é uma constante, que nos experimentos realizados assumiu valor 0.5 como valor mais apropriado para diferentes tipos de imagens avaliados. Este valor é definido empiricamente, pela observação dos resultados obtidos, onde não houveram mudanças significativas, para valores entre 0.3 e 0.6.

3. Todos os *pixels* da imagem $M_g(x, y)$ que tiverem valores maiores que o limiar T_{step} e tiverem o menor valor em $I_Z(x, y)$ com relação a seus vizinhos que tenham valor maior que T_{step} , são marcados como bordas na imagem de bordas de profundidade $B_{step}(x, y)$ resultante. Este procedimento faz com que a borda de profundidade tenha espessura de apenas 1(um) *pixel*.

Tendo identificadas as bordas de profundidade, duas direções podem ser tomadas indistintamente:

- 1ª) Continuar o processo da detecção de bordas de orientação para toda a imagem.
- 2ª) Dividir a imagem em sub-imagens, fazer o processo de detecção de bordas de orientação paralelamente para todas as sub-imagens e unir o mapa de bordas de cada sub-imagem em uma única imagem de bordas de orientação.

Estas duas direções são parte dos testes realizados neste trabalho. Tendo em vista que o desempenho do algoritmo é um ponto importante com relação aos sistemas de visão computacional, a 2ª direção é uma maneira mais otimizada de executar os processos seguintes. Neste caso, a topologia das superfícies dos objetos é mais preservada do que no processamento normal para toda a imagem, porque os processos realizados nos *pixels* de borda de uma região não sofrem influência dos *pixels* das regiões vizinhas.

Se a 1ª direção for tomada, o processo de detecção de bordas de orientação é realizado como está descrito na subseção seguinte 4.2.2.2. Se a 2ª direção for adotada, temos os seguintes passos:

1. Com um crescimento de regiões na imagem de bordas de profundidade, denominada B_{step} , é calculada uma pré-segmentação da imagem, determinando algumas regiões delimitadas por bordas de profundidade. No pior caso, pode existir apenas uma região na imagem de entrada, mesmo depois do crescimento de regiões, mas nas imagens utilizadas nos experimentos isso não aconteceu.
2. Cada região é processada em paralelo, onde são calculadas as bordas de orientação da mesma maneira como apresentado anteriormente para a 1ª direção.
3. Combina-se todos as bordas de orientação de cada uma das regiões processadas e aplica-se o refinamento do mapa de bordas e o crescimento de regiões que são apresentados nas seções 4.2.3 e 4.2.4, respectivamente.

Neste caso, o tempo para a segmentação de uma imagem de profundidade é, praticamente, o tempo de processamento da maior região da imagem, formada pelo crescimento

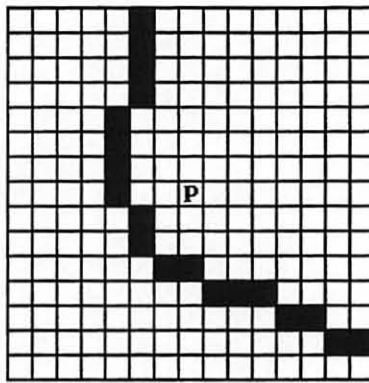
de regiões na imagem de bordas de profundidade, mais o cálculo das bordas de profundidade. O processamento paralelo, neste caso, é considerado ideal, onde as regiões podem ser processadas independentemente. Desta forma, é possível obter um segmentador de imagens de profundidade muito eficiente com relação ao tempo de execução e eficaz quanto a preservação da topologia dos objetos. Os resultados experimentais desta abordagem são mostrados no capítulo 5. Convém ressaltar que, embora a técnica de crescimento de regiões seja utilizada nessa etapa, não há a dependência de valores de limiar.

4.2.2.2 Bordas de orientação

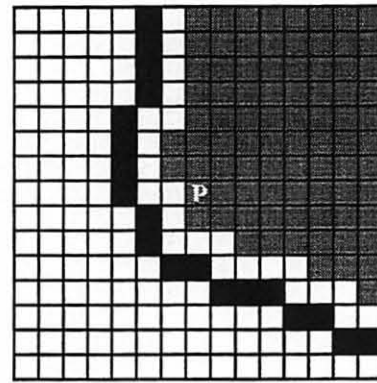
O processo de detecção de bordas de orientação segue as seguintes etapas:

1) Calcula-se os vetores normais à superfície de cada *pixel* da imagem, utilizando suas informações tridimensionais. Considere que estas informações são fornecidas pela imagem $I_{XYZ}(x, y)$, que é formada pelas 3 bandas (X, Y, Z) da imagem de profundidade. A imagem resultante deste processo é a imagem $N_v(x, y)$ dos vetores normais. Esta imagem é calculada como:

a) Para cada *pixel* $p(x, y)$ de $I_{XYZ}(x, y)$ que não seja borda em $B_{step}(x, y)$ são marcados os *pixels* válidos em uma janela de tamanho $N \times N$ centrada neste *pixel*. Primeiramente, marca-se $p(x, y)$ como válido e faz-se um crescimento de regiões nesta janela para marcar todos os *pixels* válidos com os seguintes critérios: ser um *pixel* de uma vizinhança 4 de um *pixel* válido e não ser de uma vizinhança 8 de um *pixel* de borda da imagem $B_{step}(x, y)$. Desta maneira todos os *pixels* que estiverem de um lado da borda são marcados como válidos e os do outro lado da borda são marcados como inválidos, sendo que o lado dos válidos é o mesmo lado do *pixel* central da janela. Este procedimento faz com que o cálculo do vetor normal de $p(x, y)$ leve em consideração apenas estes *pixels* válidos da janela, evitando assim que *pixels* de outras superfícies (separadas pela borda) causem distorções no processo. A figura 4.1 mostra um exemplo de uma região da imagem onde a janela, neste caso 15×15 , está sobre uma borda. Os blocos pretos são os *pixels* de borda e os *pixels* válidos são os blocos cinzas, marcados para o cálculo do vetor normal do *pixel* central $p(x, y)$.



(a) Máscara 15x15 sobre uma região de bordas da imagem $B_{step}(x, y)$



(b) *Pixel* válidos na máscara para o cálculo do vetor normal de $p(x, y)$

Figura 4.1: Resultado do processo de crescimento de regiões dos *pixels* válidos.

Para os *pixels* da imagem $I_{XYZ}(x, y)$ que são bordas na imagem $B_{step}(x, y)$, o vetor normal não é calculado e sim atribuído com o valor do vetor normal do seu vizinho mais próximo com relação à coordenada (Z), depois de calculados os vetores normais para todos os *pixels* que não são bordas. Este procedimento evita distorções nos valores das normais dos *pixels* próximos a uma borda, que dificultam uma obtenção precisa das bordas de orientação.

Um cuidado adicional deve ser levado em consideração quanto à atribuição dos *pixels* válidos em uma região como a mostrada na figura 4.2 (em destaque com um retângulo branco).

Pode ocorrer, neste caso, *pixels* válidos dos dois lados da borda pelo processo de crescimento de regiões, afetando assim a precisão do cálculo do vetor normal de $p(x, y)$. A figura 4.3 mostra o conjunto de *pixels* válidos mais apropriado para o cálculo do vetor normal de $p(x, y)$ (figura 4.3(b)) e o resultado do crescimento de regiões com os critérios apresentados anteriormente (figura 4.3(c)).

Esta situação ocorre em regiões onde a borda de profundidade não tem continuidade, formando um *pixel* de extremidade da borda. Normalmente, isto ocorre em regiões de interseção de bordas de profundidade com bordas de orientação. O *pixel* é definido como um *pixel* de borda de extremidade se na sua vizinhança 8 tenha apenas um vizinho de

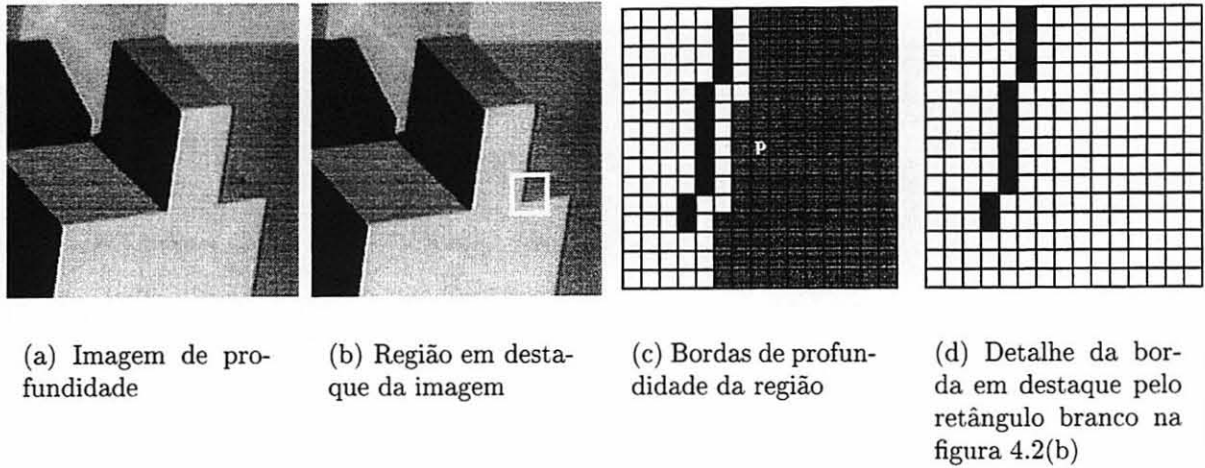


Figura 4.2: Situação onde deve ser tomado cuidado na marcação dos pixels válidos para o cálculo do vetor normal de $p(x, y)$.

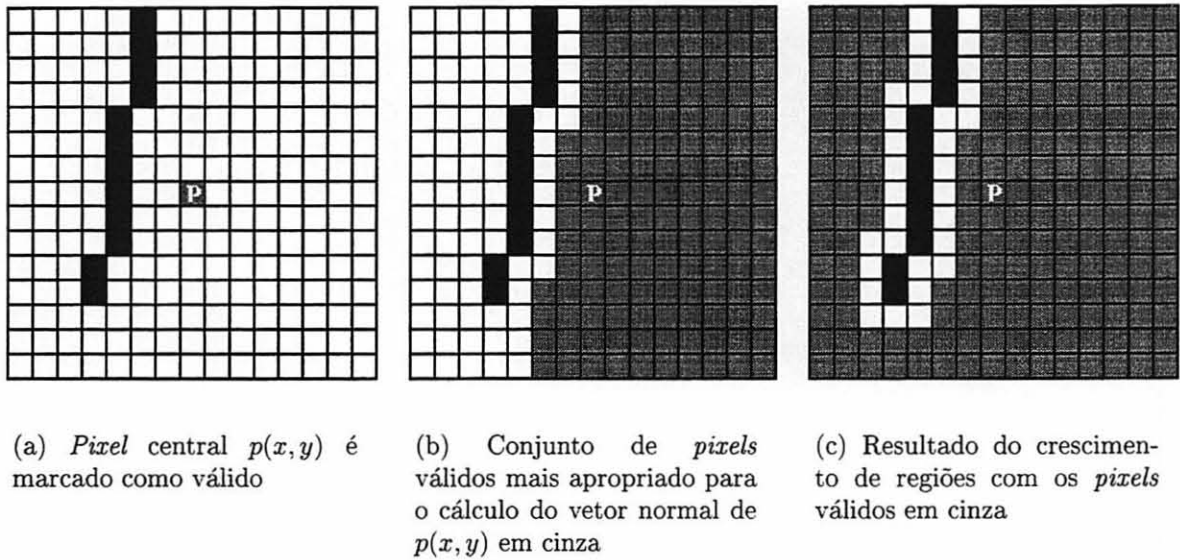


Figura 4.3: Pixels válidos marcados no processo de crescimento de regiões para o cálculo do vetor normal.

borda. Então, para o cálculo do vetor normal dos pixels $p(x, y)$ que tiverem algum pixel de extremidade na janela $N \times N$ centrada em $p(x, y)$, uma etapa é realizada antes do processo de crescimento de regiões para restringir a ação do crescimento. Nesta etapa uma borda existente por hipótese é traçada dentro da janela na direção da borda, a partir do pixel de extremidade até o limite da janela. Os pixels desta borda hipotética são uma espécie de continuidade forçada desta borda, que na verdade existe, mas é muito suave nesta região. A figura 4.4 mostra a identificação do pixel de extremidade da borda (figura

4.4(a) simbolizado por e), a direção da borda (figura 4.4(b) simbolizado por \downarrow), os *pixels* marcados na janela representando a borda hipotética (figura 4.4(c) simbolizado por x) e o resultado do crescimento de regiões dos *pixels* válidos (figura 4.4(d) em cinza).

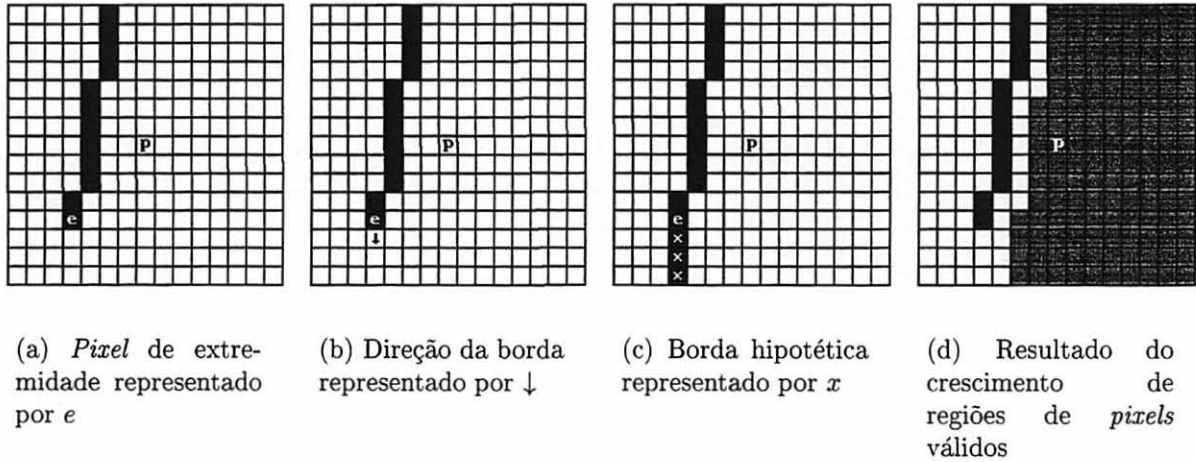


Figura 4.4: Crescimento de regiões dos *pixels* válidos para o cálculo do vetor normal em $p(x, y)$, para situações com *pixels* de borda de extremidade.

b) Este conjunto de *pixels* $M_{XYZ} = (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ válidos formado por N elementos, é usado para o cálculo da equação do plano que fornecerá os índices do vetor normal à superfície de cada *pixel* $p(x, y)$. Primeiramente, encontra-se o centro de gravidade $Centro_{XYZ}(x, y)$ dos *pixels* válidos:

$$\begin{aligned} Centro_X(x, y) &= \sum_{n=1}^N \frac{M_X(x_n, y_n)}{N} \\ Centro_Y(x, y) &= \sum_{n=1}^N \frac{M_Y(x_n, y_n)}{N} \\ Centro_Z(x, y) &= \sum_{n=1}^N \frac{M_Z(x_n, y_n)}{N} \end{aligned} \quad (4.3)$$

c) Calcula-se a matriz *scatter* equivalente, de tamanho 3x3, deste conjunto de *pixels* utilizando o centro de gravidade [13]. Nesta matriz são calculados os auto-vetores (*eigen-vectors*) baseado no método *Householder* [34]. Este método reduz a matriz esparsa original por algumas transformações ortogonais, de forma que a equação do plano com menor erro é encontrada. O vetor normal do *pixel* $p(x, y)$ é o auto-vetor com menor auto-valor (*eigen-value*). Ao final deste processo a imagem dos vetores normais $N_v(x, y)$ é obtida.

2) O gradiente e a magnitude do gradiente de $N_v(x, y)$, é calculado baseado na diferença angular entre os vetores normais, fornecendo uma imagem de gradientes $M_v(x, y)$, similar

ao gradiente $M(x, y)$ calculado anteriormente para as bordas de profundidade.

3) Calcula-se um limiar automático T_{roof} da mesma maneira que limiar T_{step} calculado para as bordas de profundidade.

4) Todos os *pixels* da imagem $M_v(x, y)$ que tiverem valores maiores que o limiar T_{roof} são marcados como bordas na imagem de bordas de orientação $B_{roof}(x, y)$ resultante. Um refinamento nesta etapa pode ser aplicado, marcando-se primeiramente os *pixels* com os maiores gradientes e depois os *pixels* não redundantes. Este processo evita que a borda fique muito grossa, dificultando o processo de afinamento posterior. Para considerar um *pixel* como redundante, podem ser utilizados elementos estruturantes de Morfologia Matemática. Nos experimentos realizados, não houve a necessidade de ser adotado este refinamento, pois as bordas mantiveram a topologia das superfícies.

A posição dos *pixels* de bordas de orientação é menos precisa do que as bordas de profundidade, devido à precisão dos sensores e do processo de suavização inicial que deixa a descontinuidade de orientação das superfícies mais suaves. Por isso, as bordas de orientação são geralmente mais grossas e mais fragmentadas, sendo necessário um processo posterior de refinamento para se obter uma melhor precisão na localização da borda.

A dificuldade de obter uma borda de orientação mais precisa também está diretamente relacionada ao cálculo dos coeficientes do vetor normal. Alguns experimentos realizados comprovaram que as superfícies dos objetos possuem ruídos mesmo depois de suavizadas, o que dificulta a obtenção de vetores normais precisos. Algumas abordagens durante este trabalho foram implementadas para a solução deste problema, entretanto, a mais eficaz foi a apresentada anteriormente.

A primeira implementação calculava os coeficientes do vetor normal através de 3 *pixels* escolhidos aleatoriamente dentro de uma janela de tamanho $N \times N$. Estes 3 *pixels* geravam uma equação do plano, podendo ser extraído os coeficientes do vetor normal. O problema estava em definir o tamanho desta janela e nem sempre os 3 *pixels* gerados aleatoriamente geravam resultados satisfatórios. Para resolver este problema, foi adotado um cálculo de estimativa robusta baseado na minimização do erro quadrático do ajuste de cada plano. Este processo gerava M conjuntos de 3 *pixels* em uma janela $N \times N$, calculava o

erro quadrático de ajuste do plano para todos os *pixels* da janela e o plano com menor erro fornecia os coeficientes do vetor normal do *pixel* central da janela. Apesar de gerar resultados melhores quanto maior o valor de M , mais lento ficava o processo, de maneira que uma solução mais rápida e eficiente precisou ser adotada. Com o cálculo dos vetores normais baseado em auto-vetores obteve-se resultados satisfatórios com um desempenho surpreendentemente melhor.

Ao combinar as imagens de bordas de profundidade $B_{step}(x, y)$ e bordas de orientação $B_{roof}(x, y)$, tem-se o mapa de bordas da imagem $I_{borda}(x, y)$ que é refinado como mostra a subseção seguinte.

4.2.3 Refinamento do mapa de bordas

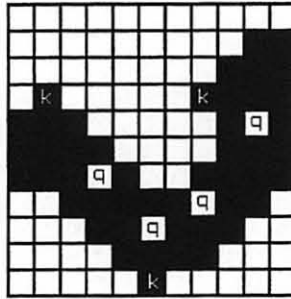
Nesta etapa um processo de afinamento é aplicado no mapa de bordas para garantir uma borda de espessura de apenas 1(um) *pixel* e depois um processo de fechamento para unir eventuais fragmentos de bordas. Estes processos são fundamentais para garantir a segmentação da imagem em regiões significativas.

Foi realizado inicialmente um levantamento de alguns métodos de afinamento [27, 38, 41] e uma análise de suas características e resultados.

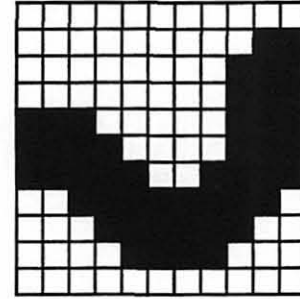
O processo de afinamento desenvolvido é um método híbrido baseado no processo de esqueletização (*skeleton*) pela transformação do eixo médio [36] e em Morfologia Matemática. Este método desenvolvido é totalmente independente de limiares e não precisa de nenhum parâmetro inicial, a não ser a imagem de entrada de bordas. Para obter este método automático, foi estudado e criado um conjunto mínimo de elementos estruturantes capaz de afinarem qualquer padrão de bordas, mantendo sua forma. A ordem com que estes elementos foram testados, também contribuiu para a criação deste método.

Primeiramente é feita a eliminação de ruídos nas bordas ocasionados por falhas no processo de detecção de bordas de orientação. Estes ruídos são: *pixels* marcados como bordas considerados parasitas e *pixels* não marcados como bordas, considerados buracos nas bordas. Os parasitas são *pixels* redundantes nas bordas que dificultam o processo de afinamento baseado na esqueletização e os buracos são *pixels* não borda que possuem na

sua vizinhança (vizinhança 4) todos os vizinhos como *pixels* de borda. A figura 4.5 mostra um exemplo de uma região de bordas com *pixels* parasitas k , *pixels* buracos q e a borda resultante depois da eliminação destes ruídos.



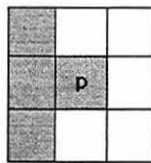
(a) Região de bordas com *pixels* com ruído



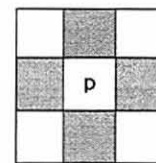
(b) Resultado da eliminação dos *pixels* de ruído

Figura 4.5: *Pixels* considerados ruídos nas bordas. Parasitas, representados por k e os buracos representados por q .

A eliminação dos ruídos nas bordas é feita utilizando dois elementos estruturantes baseados em Morfologia Matemática mostrados na figura 4.6. O primeiro (4.6(a)) elimina todos os *pixels* que tiverem esta configuração na sua vizinhança e o segundo (4.6(b)) inclui um *pixel* de borda que possui esta configuração na sua vizinhança. Este processo é aplicado uma única vez na imagem.



(a)



(b)

Figura 4.6: Elementos estruturantes para a eliminação de ruídos nas bordas.

Após a eliminação dos ruídos na borda, um processo de esqueletização do mapa de bordas é realizado da seguinte maneira:

1. Atribui valor 1 aos *pixels* na imagem do esqueleto $S(x, y)$ que são *pixels* de borda na imagem $I_{bordas}(x, y)$, ou seja, diferentes de 0(zero).

2. Aumenta em 1(um) na imagem do esqueleto $S(x, y)$ todos os *pixels* de borda em $I_{bordas}(x, y)$ que possuem todos os vizinhos (vizinhança 4) com um valor igual ao seu.
3. Considera que a imagem $S(x, y)$ é a imagem de bordas $I_{bordas}(x, y)$ e repete o passo anterior até que não haja mais mudanças no valor dos *pixels* da imagem $S(x, y)$. Desta forma os *pixels* centrais da borda têm valores maiores do que os pixels periféricos.
4. Remove os *pixels* da imagem, que tenham na sua vizinhança (vizinhança 8) um vizinho com valor maior que o seu, por um processo de erosão a partir dos *pixels* com menor valor. Este processo se repete até não existirem *pixels* a serem removidos. Um cuidado adicional deve ser mantido para que a borda não seja fragmentada, para isso são utilizados alguns elementos estruturantes baseados em Morfologia Matemática que garantem que um *pixel* candidato a ser eliminado não cause um fragmento na borda.

A figura 4.7 mostra o processo de esqueletização do mapa de bordas. Na figura 4.7(c) é mostrado o valor de cada *pixel* depois do processo de atribuição de valores aos *pixels*, onde os mais internos possuem valores maiores. A figura 4.7(d) é o resultado do processo de esqueletização.

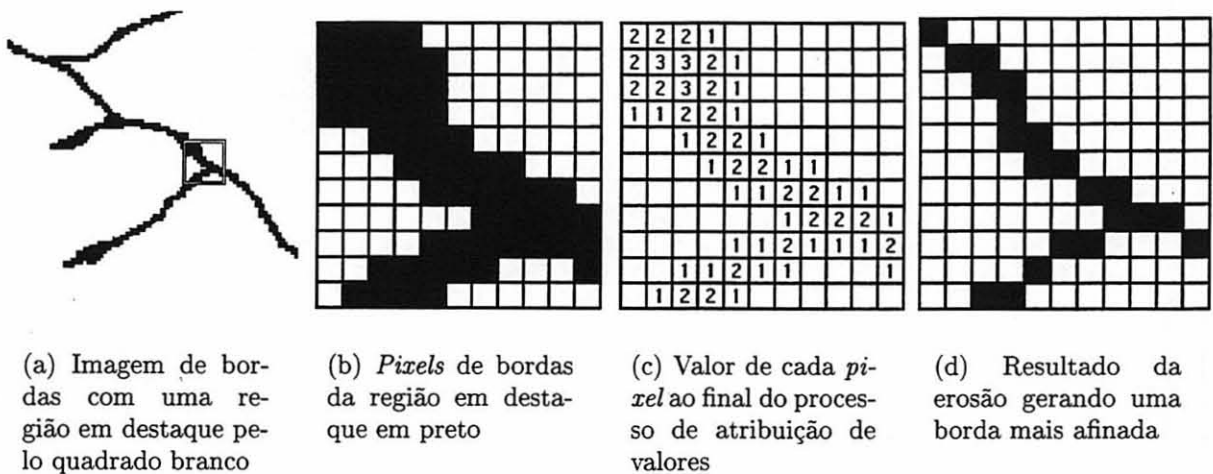


Figura 4.7: Processo de esqueletização do mapa de bordas

A grande maioria dos algoritmos de esqueletização não preserva totalmente a topologia das bordas, principalmente nas extremidades, pois utiliza vizinhança 4 para o processo de erosão para evitar a fragmentação do mapa de bordas. Isso dificulta alguns processos como por exemplo o fechamento das bordas, pois aparecem alguns fragmentos de bordas diferentes da forma original da borda. As diferenças dos processos de esqueletização com erosão utilizando vizinhança 4 e 8 estão mostrados na figura 4.8.

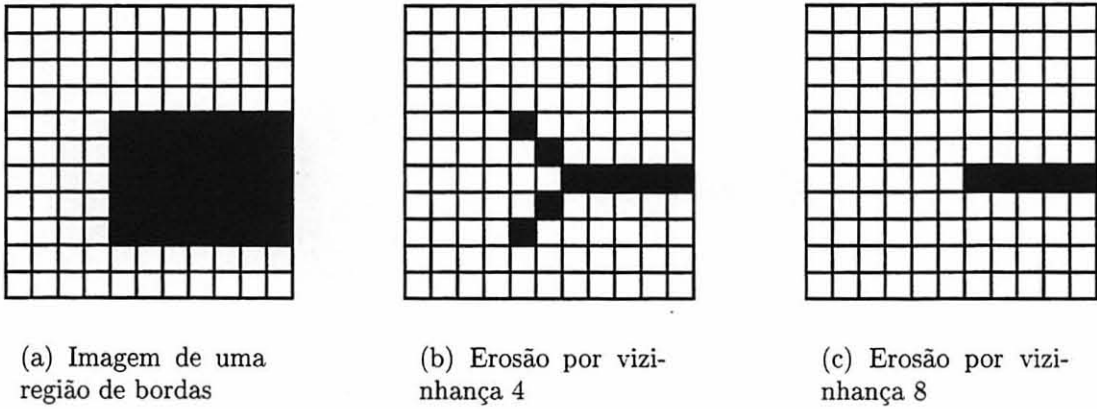


Figura 4.8: Diferença da erosão por vizinhança 4 e por vizinhança 8 no processo de esqueletização.

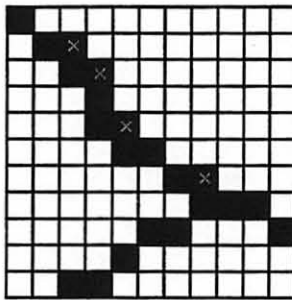
Para contornar este problema, foi utilizada a vizinhança 8 para o processo de erosão e um elemento estruturante baseado em Morfologia Matemática para evitar a fragmentação da borda. Se um *pixel* $p(x, y)$ candidato a ser removido e seus vizinhos tiverem a configuração dos elementos estruturantes, então este *pixel* não é removido. A figura 4.9 mostra os elementos estruturantes utilizados neste processo. São utilizadas as 4 variações do elemento (0, 90, 180, 270 graus) da figura 4.9(a) e as 2 variações do elemento (0 e 90 graus) da figura 4.9(b).



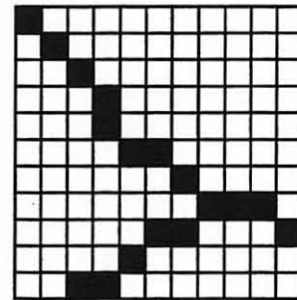
Figura 4.9: Elementos estruturantes utilizados no processo de erosão.

No esqueleto do mapa de bordas (4.7(d)) é possível notar que ainda existem *pixels*

redundantes que podem ser afinados para que a espessura da borda tenha apenas 1 (um) *pixel* (figura 4.10). Estes *pixels* são eliminados por um processo de afinamento baseado em elementos estruturantes. Todos os *pixels* $p(x, y)$ que na sua vizinhança não tiverem a configuração dos elementos da figura 4.9 e os que tiverem a configuração dos elementos da figura 4.11 são removidos. Neste caso também são consideradas as variações de cada elemento estruturante. Este processo é realizado até que não existam mais *pixels* a serem eliminados.

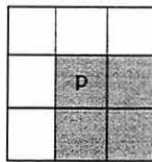


(a) Bordas com *pixels* redundantes x

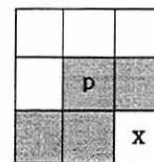


(b) Bordas afinada sem os *pixels* redundantes

Figura 4.10: Região de bordas do resultado do processo de esqueletização com *pixels* redundantes.



(a)



(b)

Figura 4.11: Elementos estruturantes utilizados no processo de eliminação de *pixels* redundantes.

Com o mapa de bordas afinado, um processo de fechamento é aplicado para eliminar os eventuais fragmentos entre as bordas. Este fragmentos são muito comuns nas regiões de interseção de bordas de profundidade e bordas orientação. Estes *pixels* não são marcados como bordas de profundidade por possuírem um gradiente muito pequeno de profundidade, da mesma forma que não são marcados como *pixels* de bordas de orientação por terem

um gradiente angular baixo. O processo de esqueletização em alguns casos pode afinar a extremidade das bordas, aumentando o fragmento entre as bordas.

O processo de fechamento é composto das seguintes etapas:

1. Cada *pixel* de extremidade $e(x, y)$ de borda da imagem é posicionado no centro de uma janela de tamanho $N \times N$.
2. O partir de cada $e(x, y)$ é feita uma varredura em raio para saber se existe algum *pixel* de borda próximo, descartando os *pixels* de borda que pertencem ao mesmo segmento de $e(x, y)$. Esta varredura é feita inicialmente num raio de 2 *pixels* até $\frac{N}{2}$.
3. Encontra-se o *pixel* de borda que tiver a menor diferença de profundidade na imagem $I_Z(x, y)$ em relação a profundidade de $e(x, y)$ e traça uma borda entre os dois.

Este procedimento evita que bordas muito próximas de diferentes objetos sejam fechadas como mostra a figura 4.13. As figura 4.12 e 4.13 mostram exemplos deste processo.

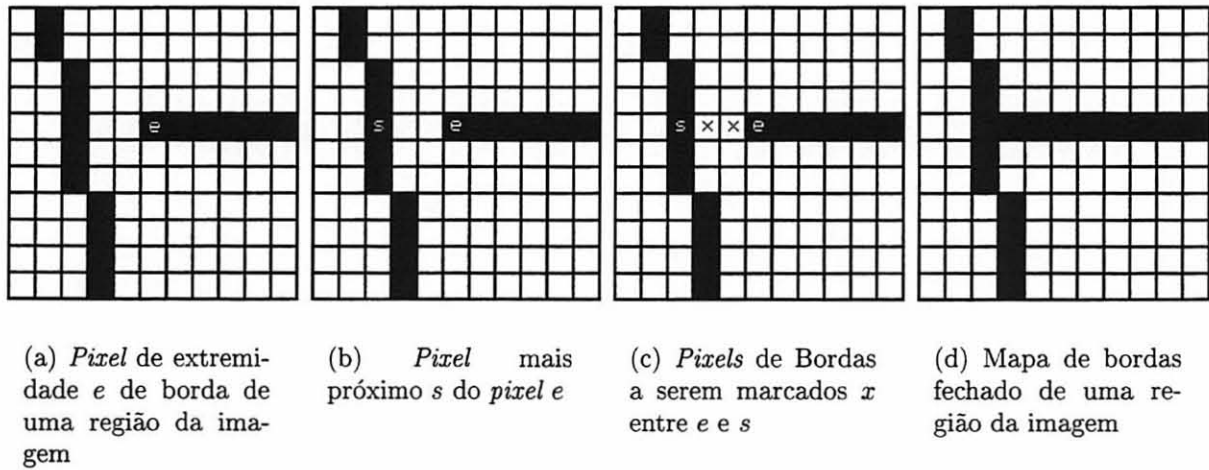


Figura 4.12: Processo de fechamento do mapa de bordas de uma região com um *pixel* de extremidade.

Ao final do refinamento, o mapa de bordas da imagem está fechado fornecendo uma imagem $I_{edge}(x, y)$, que delimita todas as regiões da imagem, ou seja a segmentação por detecção de bordas. O tamanho da janela utilizada não deve ser muito grande para evitar o fechamento de *pixels* muito distantes, podendo criar uma falsa superfície. O objetivo do fechamento é garantir que as bordas de profundidade e as bordas de orientação sejam

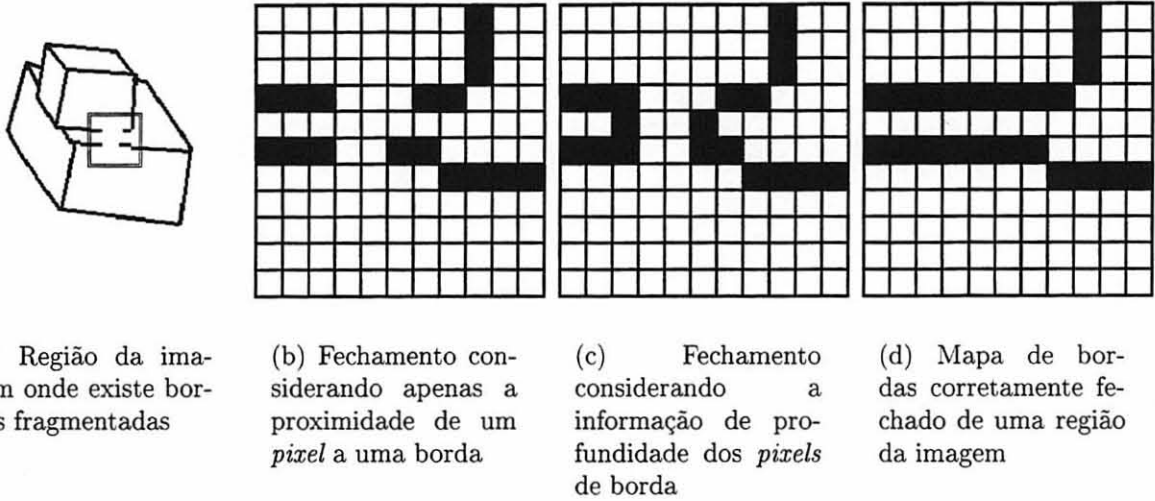


Figura 4.13: Processo de fechamento do mapa de bordas de uma região com vários *pixels* de extremidade.

fechadas e geralmente elas estão separadas no máximo por uma distância de 3(três) *pixels*. Neste caso, uma janela de tamanho 9x9 é suficiente. Este processo possui algumas limitações que poderiam ser evitadas com processos mais eficientes de fechamento, porém sem grandes diferenças nos resultados finais. Como os fragmentos são muito pequenos, esta abordagem foi suficiente para resolver o problema do mapa de bordas fragmentado. Um outra abordagem pode ser utilizada, baseada no cálculo da interseção das retas do *pixel* de extremidade e do *pixel* de borda mais próximo, de maneira a preservar ainda mais a topologia da borda nestas regiões onde ocorrem os fragmentos. Bordas seriam traçadas dos dois *pixels* (extremidade e o de borda mais próximo) até o *pixel* de interseção dessas retas e não em uma linha reta entre os dois *pixels*.

Pequenas regiões de ruído podem ser formadas pelo processo de fechamento. Normalmente estas regiões ocorrem em regiões de bordas de orientação mais fragmentadas e mal definidas. Estas regiões são removidas no processo seguinte de crescimento de regiões que fornecerá uma segmentação final da imagem com regiões mais significativas.

4.2.4 Crescimento de regiões

O crescimento de regiões é realizado para gerar a segmentação final, eliminando regiões de ruídos, garantindo uma segmentação com as regiões mais significativas da imagem e

preservando a topologia das superfícies dos objetos. Este processo divide a imagem em regiões delimitadas por bordas da seguinte maneira:

1. Marca-se, a partir da posição $I_{edge}(0, 0)$ da imagem, o primeiro *pixel* na imagem $I_{edge}(x, y)$ que não é borda como o *pixel* inicial da região.
2. Faz-se o crescimento da região unindo recursivamente cada *pixel* que não é um *pixel* de borda em $I_{edge}(x, y)$ e que é vizinho (vizinhança 4) de um *pixel* da região.
3. Repete-se os passos anteriores para novas regiões, até não existirem mais *pixels* a serem crescidos.

No final deste processo os *pixels* de bordas são agregados às regiões vizinhas que sejam mais próximas com relação à profundidade obtida da imagem $I_Z(x, y)$ e ao ângulo do vetor normal à superfície neste ponto obtido por $N_v(x, y)$.

Para eliminar as regiões pequenas, utiliza-se um limiar T_r . Todos os *pixels* de uma região que tem um número de *pixels* menor que T_r , são distribuídos para as regiões vizinhas. Esta distribuição é feita primeiramente para os *pixels* mais semelhantes com relação a profundidade e orientação de seus vizinhos que estão em outra região. Desta maneira, as regiões mais semelhantes vizinhas à região a ser removida receberão um número maior de *pixels*. Este processo mantém a topologia das superfícies agregando *pixels* eliminados de pequenas regiões vizinhas e fornece uma segmentação mais adequada para facilitar a interpretação da imagem.

Este limiar T_r pode ser definido de maneira automática, baseado em uma percentagem do tamanho das maiores regiões. Por exemplo, este limiar pode ser definido como 5% do tamanho da maior região (ou da média das 2 regiões maiores). Se a maior região tiver 1000 *pixels*, uma região com menos de 50 *pixels* será eliminada.

4.3 Avaliação

O algoritmo desenvolvido foi testado e comparado com os demais algoritmos apresentados no capítulo 3. Os resultados são apresentados no capítulo seguinte.

Com relação ao desempenho, este algoritmo é relativamente rápido quando comparado aos algoritmos WSU, UE e USF, mas pode ser otimizado se for utilizado o processamento paralelo para cada região delimitada pelo processo de detecção de bordas de profundidade.

Os limiares automáticos ajudam no fornecimento de um segmentador mais genérico para diferentes tipos de imagens e evitam um processo exaustivo de ajuste de limiares fixos.

O processo de filtragem e da eliminação das regiões pequenas, forneceram uma segmentação ideal para processos posteriores. As superfícies dos objetos tiveram suas topologias preservadas o que facilita a representação e interpretação dos objetos da imagem.

O processo de afinamento se mostrou muito eficiente para diferentes configurações de bordas, pelo fato da independência de parâmetros ou limiares, geralmente empregados em outros algoritmos de afinamento.

A limitação deste algoritmo está no cálculo dos coeficientes do vetor normal para as bordas de orientação, que é muito sensível a ruídos. Isso gera uma borda de orientação fragmentada e com ruídos. Um processo posterior de refinamento próprio para o mapa de bordas de orientação vem sendo estudado para melhor identificar a direção e localização destas bordas. Com isso, talvez o processo de afinamento do mapa de bordas final possa ser eliminado.

4.4 Conclusão

Os objetivos deste algoritmo foram alcançados e os resultados mostraram que a segmentação fornece as regiões mais significativas da imagem, com a eliminação de grande parte das regiões de ruídos.

Os testes deste algoritmo usando a abordagem do processamento paralelo mostraram um ganho no desempenho, principalmente se for realizado em um ambiente distribuído. Tanto o processo de detecção de bordas de orientação quanto o processo de crescimento de regiões podem ser implementados totalmente em paralelo.

Se for considerada a aplicação deste algoritmo para imagens com superfícies curvas, nenhuma modificação é necessária. Com a obtenção do mapa de bordas da imagem, to-

das as regiões planares foram encontradas e outras regiões com superfícies curvas foram delimitadas por bordas. Um processo posterior avalia cada região encontrada na imagem segmentada, tanto no final do processamento normal como no processamento paralelo. Cada região cujo ajuste planar tiver um erro quadrático maior que um limiar T_j , é considerada uma região de superfície curva que pode ser segmentada. Se for utilizada uma aproximação por planos nestas superfícies, um algoritmo de agrupamento como o *K-means* (seção 2.10.2) pode ser empregado para segmentar esta região. Os parâmetros do algoritmo seriam os coeficientes (a, b, c) dos vetores normais da região, o número de iterações e o número de segmentos que se deseja obter. Os vetores normais já estão calculados anteriormente, o número de iterações pode ser um limiar fixo ($T_c = 50$) e o número de segmentos S pode ser calculado como:

$$S = \frac{\max[A_r(x, y)] - \min[A_r(x, y)]}{T_\theta} \quad (4.4)$$

Onde:

$A_r(x, y)$ é a imagem de ângulos

$\max[A_r(x, y)]$ é o maior ângulo da imagem de ângulos da região (4.5)

$\min[A_r(x, y)]$ é o menor ângulo da imagem de ângulos da região

T_θ é o valor de angular mínimo entre cada segmento

A imagem de ângulos $A_r(x, y)$ pode ser calculada baseada na diferença angular entre os vetores normais da imagem $N_v(x, y)$ em um vetor referência qualquer no espaço 3D.

Quanto menor T_θ , maior o número de segmentos.

Desta forma é possível obter um segmentador de imagens de profundidade que segmenta também superfícies curvas aproximando-as por planos. Neste processo também pode ser utilizada a abordagem do processamento paralelo para otimizar o tempo de processamento. Esta abordagem foi apresentada por Bellon et al. [5] recentemente.

CAPÍTULO 5

RESULTADOS EXPERIMENTAIS

Neste capítulo são apresentados os resultados experimentais avaliados no trabalho de Hoover et al. [21], os resultados e avaliação do método desenvolvido baseado em detecção de bordas, e os resultados comparativos do método desenvolvido com outros métodos. Ao final deste capítulo é apresentada uma breve conclusão.

5.1 Avaliação dos algoritmos da literatura

Os resultados apresentados no trabalho de Hoover mostraram que nenhum dos 4 algoritmos (USF, WSU, UE e UB) tiveram um desempenho perfeito com uma tolerância $T = 1.0$ que representa 100% de acerto na segmentação das regiões das imagens ABW, que não foi nenhuma surpresa.

Entre a tolerância de 50% a 89% os algoritmos tiveram uma boa média de acerto das regiões corretamente segmentadas. No geral, com uma tolerância moderada de 80% todos os algoritmos tiveram uma média equivalente de regiões corretamente segmentadas para as imagens ABW. Nenhum deles teve bons resultados com uma tolerância maior que 90%.

Para o levantamento desses dados, as equipes utilizaram as 10 imagens de treinamento ABW para o ajuste de seus algoritmos e a definição de seus limiares. Após o treinamento e ajuste de limiares, os algoritmos foram aplicados a cada uma das 30 imagens testes, fornecendo 30 imagens segmentadas para a comparação de resultados. Os limiares utilizados para os 4 algoritmos estão descritos no trabalho de Hoover et al. [21].

A ferramenta de comparação foi então utilizada com diferentes tolerâncias para cada uma das imagens segmentadas de cada um dos 4 algoritmos. Os parâmetros de entrada desta ferramenta para cada imagem teste foram:

1. A imagens segmentadas manualmente com os valores de ângulos entre as superfícies.

2. A imagem segmentada pelo algoritmo.

3. A tolerância $0.5 < T < 1.0$.

A média dos resultados de todas as 30 imagens testes de cada algoritmo foi utilizada para a criação de tabelas comparativas para cada tolerância. Os resultados da porcentagem de cada tipo de classificação (sobre-segmentação, sub-segmentação, ruído e perdidos) de cada região foram também comparados.

No geral, os resultados mostraram que a predominância de regiões sobre-segmentadas é maior do que regiões sub-segmentadas e a predominância das regiões perdidas é maior que as regiões classificadas como ruídos. O tempo de processamento do algoritmo UB foi consideravelmente menor que os demais e o algoritmo WSU foi dramaticamente maior.

O problema na comparação dos resultados está na identificação de quais são as propriedades das superfícies que geram uma segmentação incorreta e quais são os tipos de erros encontrados. As comparações mostram que regiões grandes são segmentadas mais precisamente que regiões pequenas. Nas regiões pequenas existe uma grande quantidade de erros, tanto no processo de aquisição da informação de profundidade quanto no resultado de sua segmentação. E muitas vezes estas pequenas regiões são eliminadas ou não se preserva sua topologia.

A grande maioria das aplicações em processamento de imagens tem um interesse predominante em regiões grandes, de forma que os algoritmos de segmentação existentes tem um desempenho suficientemente bom neste caso. Entretanto, para aplicações que necessitem de uma correta segmentação que preserve também as pequenas regiões, os algoritmos de segmentação necessitam de melhoramentos, a começar pela segmentação de superfícies planares que ainda não está definitivamente resolvida.

Um ponto importante está no desempenho ruim dos algoritmos na precisão das regiões corretamente detetadas quando aplicados a uma tolerância maior que 90%. Isto indica que a precisão na bordas das regiões necessita de refinamentos para garantir uma correta descrição topológica das superfícies.

Quanto à generalização dos algoritmos, a grande maioria tem uma dependência grande de limiares. Muitos deles sofrem grandes variações para tipos diferentes de imagens,

dificultando assim a obtenção de um algoritmo genérico que possa ser empregado em diferentes aplicações.

A seção seguinte mostra os resultados de cada etapa do método desenvolvido e sua avaliação. Os resultados comparativos entre os algoritmos e o métodos desenvolvido será apresentado na seção 5.3 deste capítulo.

5.2 Avaliação do método desenvolvido

Os resultados do método de segmentação desenvolvido baseado em detecção de bordas, mostraram alguns ganhos com relação aos problemas levantados neste trabalho e no trabalho de Hoover et al. [21] para a segmentação de imagens de objetos poliédricos.

A segmentação é mais precisa com relação à descrição topológica das superfícies e as superfícies pequenas significativas na imagem são preservadas. A eliminação de superfícies ruidosas fornece uma segmentação mais definida que facilita processos seguintes como a representação, interpretação e o reconhecimento de objetos.

O método possui uma dependência menor de limiares facilitando sua aplicação para diferentes tipos de imagens. Os parâmetros de calibração de limiares necessários foram:

1. Na filtragem inicial foi utilizado um filtro mediano de tamanho 3x3 aplicado uma vez nas 3 bandas (X, Y, Z) da imagem.
2. O cálculo dos limiares das bordas de profundidade T_{step} e orientação T_{roof} foram feitas de forma automática baseado na informação de gradiente. As constantes C_{step} e C_{root} deste processo assumiram valores 0.5.
3. O tamanho da janela utilizada no cálculo dos coeficientes dos vetores normais no processo de detecção de bordas de orientação foi de 19x19. Uma janela de tamanho menor causou muitas bordas falsas, porque a imagem de profundidade ABW é muito ruidosa e uma janela maior além de engrossar demais as bordas de orientação, causou perdas de precisão em regiões muito pequenas.
4. No refinamento do mapa de bordas foi utilizado uma janela de tamanho 9x9 para o

processo de fechamento. Como os fragmentos identificados que precisam ser fechados não ultrapassam o tamanho de 3 *pixels*, este tamanho de janela foi suficiente.

5. O limiar T_r utilizado para eliminação das regiões pequenas no processo de crescimento de regiões assumiu um valor automático, baseado em uma percentagem do tamanho da maior região da imagem. Todas as regiões com tamanho menor que 0.1% do tamanho da maior região da imagem são eliminadas.

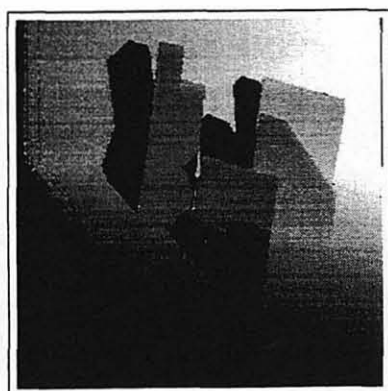
A seguir será mostrada cada etapa do método desenvolvido com a imagem intermediária resultante de cada etapa na figura 5.1. Neste processo foi utilizada a abordagem por processamento paralelo. Este processamento paralelo foi realizado no aplicativo KHOROS¹ que disponibiliza uma plataforma para processamento de aplicações em um ambiente distribuído. O aplicativo KHOROS foi utilizado para desenvolvimento por ter uma versão estudantil gratuita e por ser um pacote aberto com predominância da linguagem C.

A figura 5.1(a) mostra uma imagem de profundidade utilizada nos testes, com sua respectiva imagem de intensidade luminosa 5.1(b). Nesta imagem é possível identificar ruídos impulsivos e regiões sem informação nas regiões de sombras. A imagem segmentada manualmente para comparações é mostrada na figura 5.1(c).

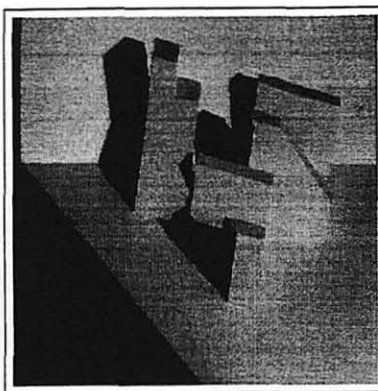
O processo de filtragem é mostrado na figura 5.1(d), onde foi utilizado um filtro mediano de tamanho 3x3 aplicado uma vez. Nos testes realizados foi possível identificar que a maioria dos ruídos eram impulsivos e um filtro mediano era o mais apropriado para a filtragem. Quanto ao tamanho do filtro, o tamanho ideal foi 3x3, um tamanho maior causou na grande maioria dos testes uma borda de orientação mais espalhada e contornos mais arredondados nos objetos.

O processo de detecção de bordas de profundidade é mostrado na figura 5.1(e), onde pode-se identificar que a maioria das bordas têm apenas um *pixel* de espessura, não necessitando de refinamento, o que facilita os processos seguintes. Regiões com grande quantidade de ruído ou falta de informação foram identificadas e eliminadas no processo seguinte de crescimento de regiões. O crescimento de regiões, mostrado na figura 5.1(f),

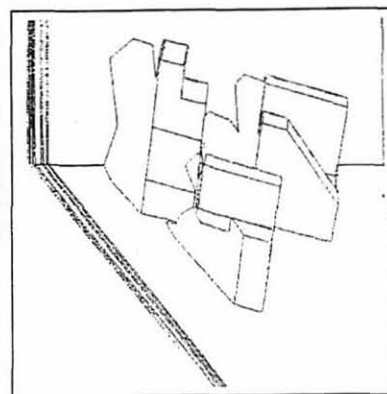
¹O aplicativo KHOROS pode ser obtido no *site* <http://www.khoros.com>



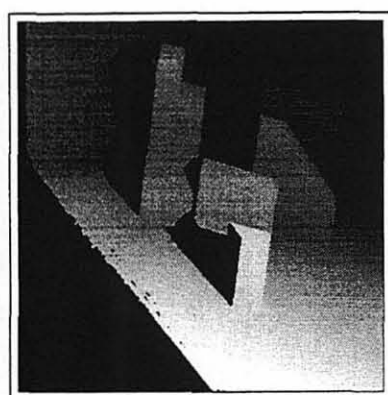
(a) Imagem de profundidade



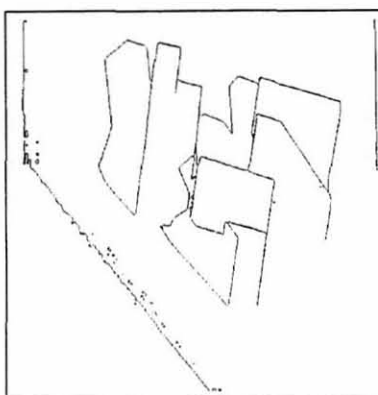
(b) Imagem de intensidade



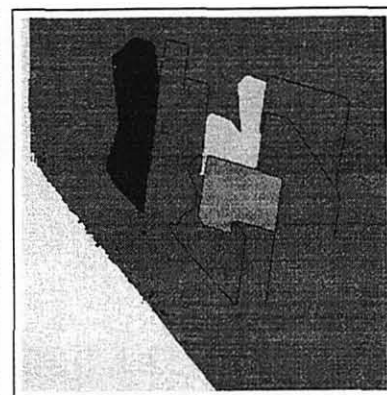
(c) Segmentação manual



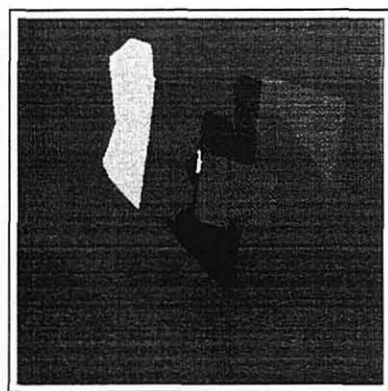
(d) Imagem filtrada



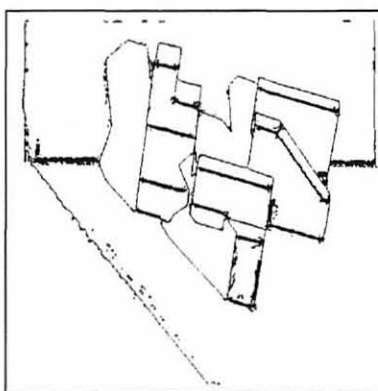
(e) Bordas de profundidade



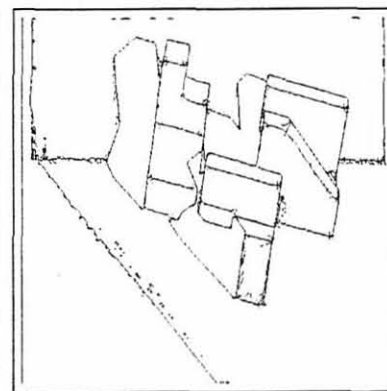
(f) Crescimento de regiões na imagem de bordas de profundidade



(g) Eliminação das bordas e regiões pequenas



(h) Mapa de bordas



(i) Mapa de bordas afinado

foi realizado para delimitar regiões para o processamento paralelo das etapas seguintes. Foram eliminadas as bordas e as regiões pequenas, restando apenas as regiões mais significativas visualizadas na figura 5.1(g).

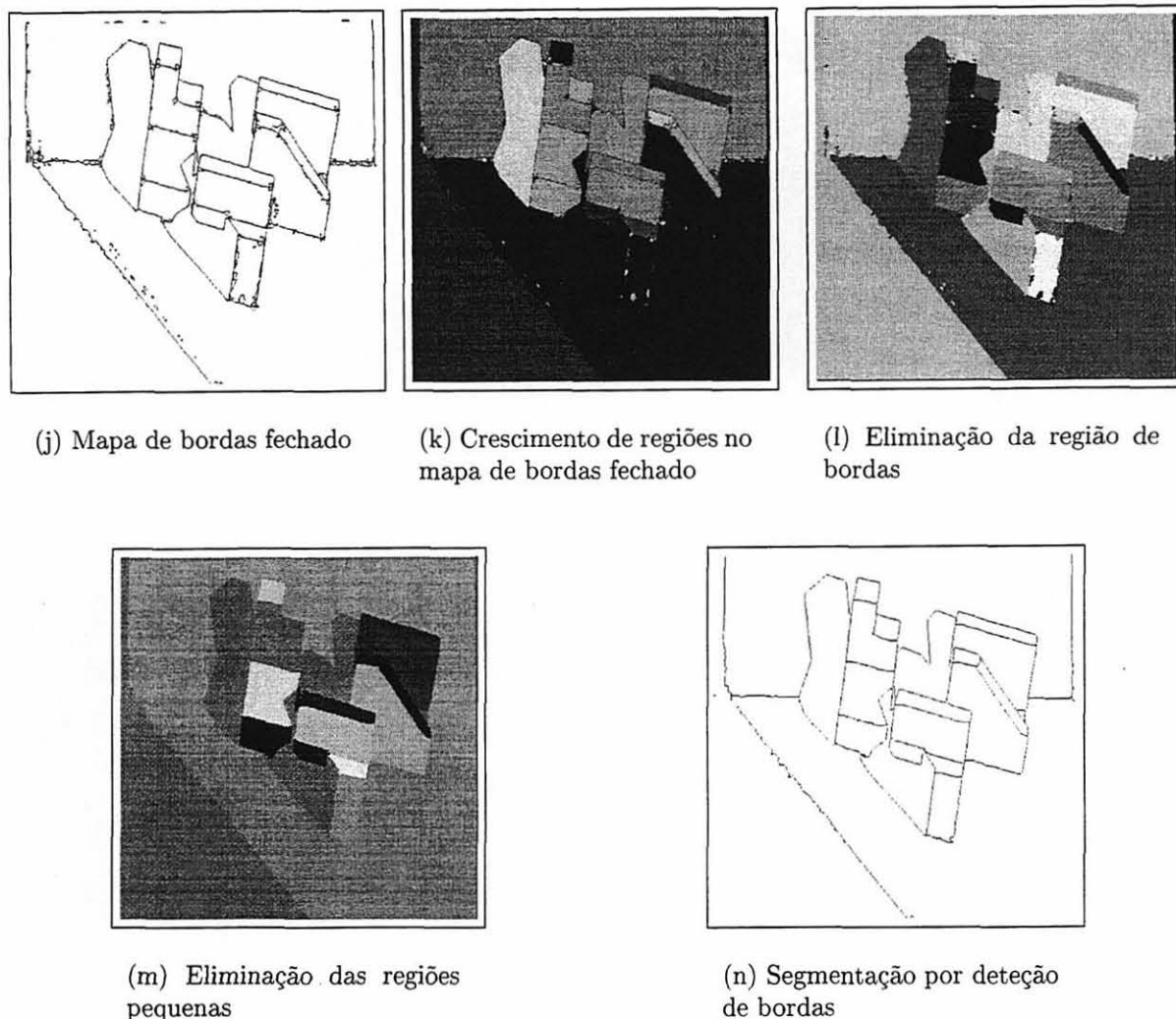


Figura 5.1: *Etapas do processo de segmentação de imagens de profundidade por detecção de bordas em uma imagem de profundidade do tipo ABW.*

O cálculo dos coeficientes dos vetores normais às superfícies dos *pixels* de cada região delimitada e a detecção das bordas de orientação foram realizados em paralelo, de maneira a otimizar o tempo de processamento do método.

A união dos mapas de bordas de profundidade e orientação é apresentada na figura 5.1(h). Este mapa de bordas possui bordas com espessura grossa e fragmentos que precisam ser tratados. O refinamento das bordas é realizado para obter uma borda com espessura de apenas um *pixel* como mostra a figura 5.1(i). Esta etapa mostra que a forma da borda foi preservada e a presença de bordas falsas é devido a qualidade do resultado das bordas de orientação. Esta deficiência é resolvida com os processos seguintes. Nota-se que existem falhas e fragmentos de bordas no mapa de bordas, principalmente em regiões

de interseção de bordas de profundidade e bordas de orientação. O fechamento é realizado para obter um mapa de bordas significativo que forneça a segmentação das regiões delimitadas por bordas. O resultado do fechamento é mostrado na figura 5.1(j).

Com o fechamento, obtém-se a segmentação por detecção de bordas, porém a topologia dos objetos deve ser preservada com um refinamento desse mapa de bordas. O refinamento é realizado com a remoção das regiões pequenas, regiões de bordas e regiões de ruídos na imagem resultante de um processo de crescimento de regiões. O crescimento de regiões no mapa de bordas fechado, a remoção das bordas e das regiões pequenas é mostrado nas figuras 5.1(k), 5.1(l) e 5.1(m) respectivamente. Nota-se nesta etapa que as regiões eliminadas contribuíram para a preservação da topologia das superfícies dos objetos, facilitando a interpretação da imagem. Regiões pequenas tiveram suas características preservadas, como não foi observado com outros segmentadores. Ao final do processo pode-se representar a segmentação final por bordas como mostra a figura 5.1(n).

A figura 5.2 mostra o resultado da aplicação do método desenvolvido aplicado a uma imagem com um objeto não poliédrico [4]. Este exemplo mostra que o método não se aplica somente a superfícies planares, mas também a superfícies curvas, fornecendo uma segmentação com as regiões significativas da imagem. É possível notar que a preservação da topologia das superfícies é mantida e as regiões de ruído são eliminadas.

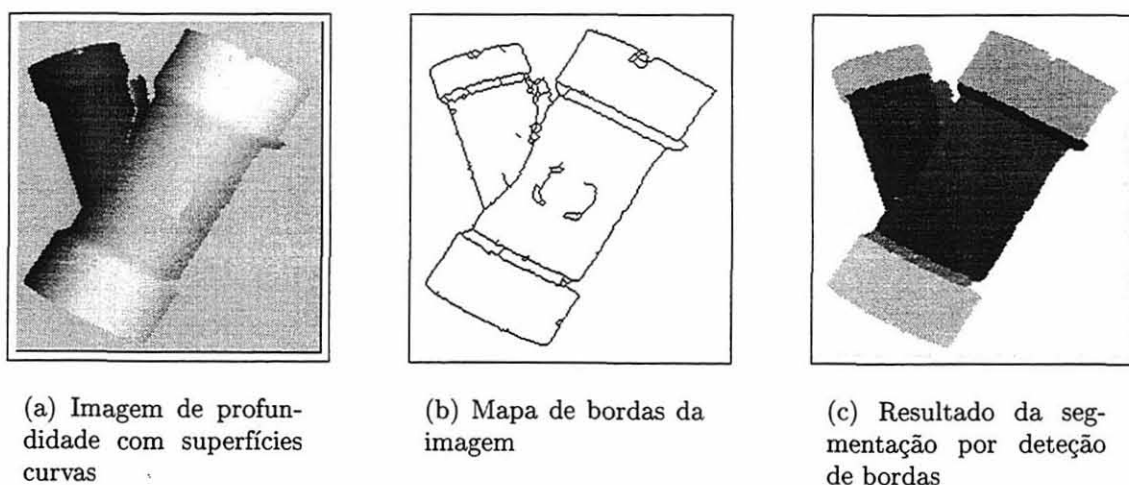


Figura 5.2: Exemplo de segmentação de uma imagem de profundidade com um objeto não poliédrico baseado na detecção de bordas.

Esta imagem é uma imagem real obtida por um sensor chamado *MSU's Technical Arts*

Região	Normal	Simultâneo	Distribuído
1	-	5:11	4:05
2	-	1:38	1:06
3	-	1:57	1:10
4	-	0:46	0:20
5	-	0:50	0:22
6	-	0:48	0:12
7	-	0:51	0:13
8	-	0:22	0:10
9	-	0:20	0:10
Total	8:30	6:12	4:37

Tabela 5.1: Tabela de desempenho do algoritmo aplicado no processamento normal na imagem inteira, simultâneo para todas as regiões e distribuído para todas as regiões.

100X Range Scanner².

A tabela 5.1 mostra o desempenho (em minutos) do algoritmo com relação ao tempo de processamento obtido nos testes com diferentes abordagens. A imagem da figura 5.1 foi submetida ao processamento normal para toda a imagem, ao processamento simultâneo para todas as regiões identificadas inicialmente e distribuído (paralelo) para todas as regiões identificadas.

Nesta imagem são identificadas 9 regiões a partir do mapa de bordas de profundidade. No processamento normal, as etapas do algoritmo são realizadas uma a uma na imagem toda em uma mesma máquina. O processamento simultâneo é realizado nas etapas a partir da identificação das regiões pelas bordas de profundidade. Neste caso, as regiões delimitadas são processadas simultaneamente em uma mesma máquina. Finalmente o processamento distribuído (paralelo) é semelhante ao simultâneo, porém cada região delimitada é processada em uma máquina diferente.

Os valores baixos para as regiões de rótulos de 4 a 9, são porque elas são regiões de sombra, onde não existe informação de profundidade na imagem. O tempo total do processamento distribuído é aproximadamente o tempo de processamento do crescimento de regiões inicial baseado na detecção de bordas de profundidade, mais o tempo de processamento da maior região. Este exemplo comprova um aumento no desempenho do algoritmo na aplicação do processamento simultâneo e distribuído do método desenvolvido.

²Estas imagens podem ser obtidas em (<http://sampl.eng.ohio-state.edu/~sampl/data/3DDB/RID/>)

Os testes para a geração desses dados foram feitos em máquinas com processador K6-350Mhz, 32Mb de memória RAM, placas de rede de 100Mbits e sistema operacional Linux.

5.3 Resultados comparativos

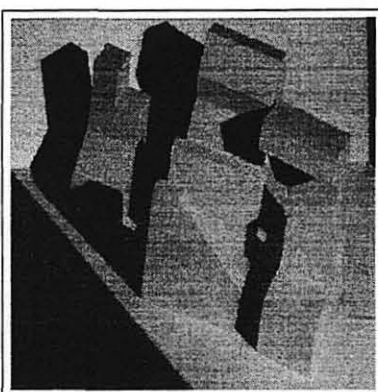
A seguir são mostradas algumas imagens de profundidade ABW utilizadas nos experimentos, divididas nas duas categorias: imagens de treinamento e imagens teste. Cada uma com sua respectiva imagem de intensidade luminosa, segmentação manual *ground truth*, segmentação por detecção de bordas e a segmentação dos 4 algoritmos (WSU, UB, UE e USF) respectivamente.

Os resultados experimentais de cada um dos algoritmos são apresentados para 6 das 10 imagens de treinamento e 6 das 20 primeiras imagens teste. Os resultados de todas as imagens aplicadas aos 4 algoritmos (WSU, UB, UE e USF) e a segmentação por detecção de bordas podem ser encontrados no *site* do grupo de pesquisa de Visão Computacional e Processamento de Imagens (IMAGO)³. Para melhor visualizar os resultados, a segmentação é apresentada com a sua representação por bordas.

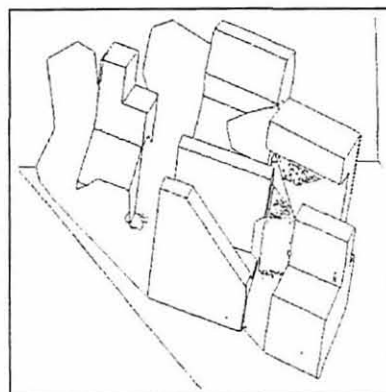
³<http://www.inf.ufpr.br/~olga/projetos>



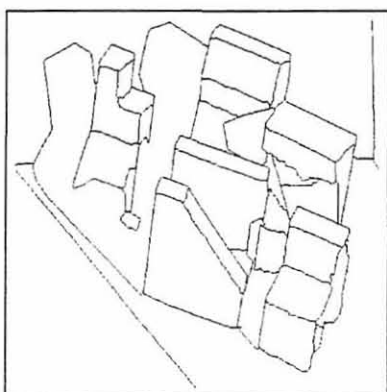
(a) Imagem de profundidade



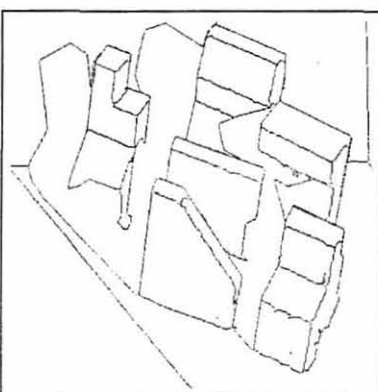
(b) Imagem de intensidade



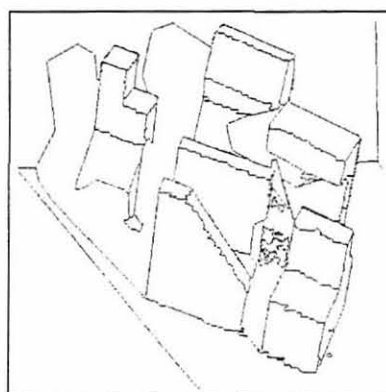
(c) Segmentação manual



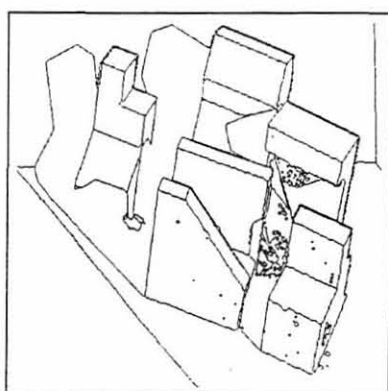
(d) Segmentação por detecção de bordas



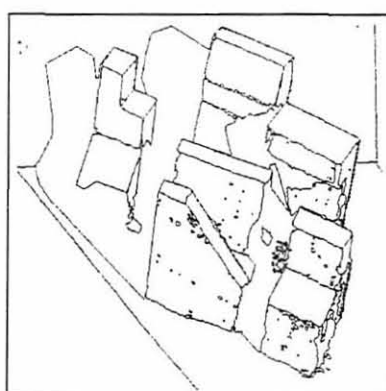
(e) WSU



(f) UB



(g) UE

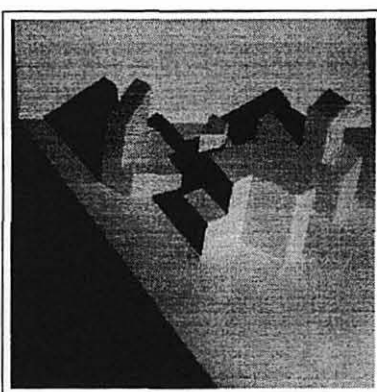


(h) USF

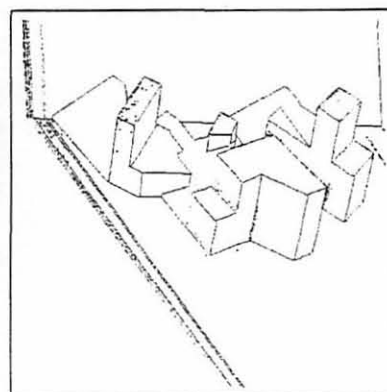
Figura 5.3: Imagem de treinamento ABW (*abw.train.0*).



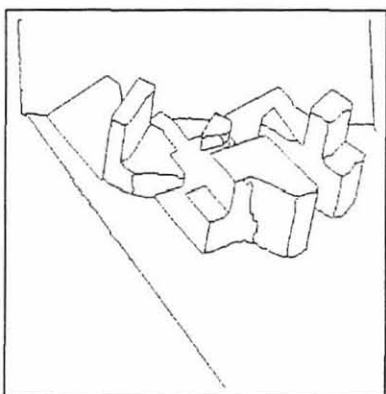
(a) Imagem de profundidade



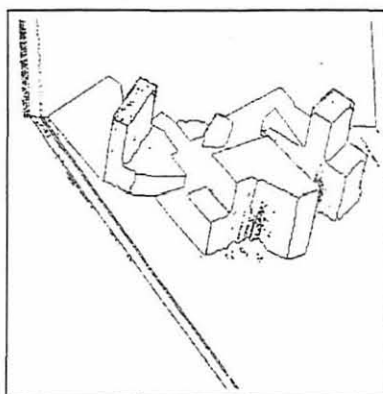
(b) Imagem de intensidade



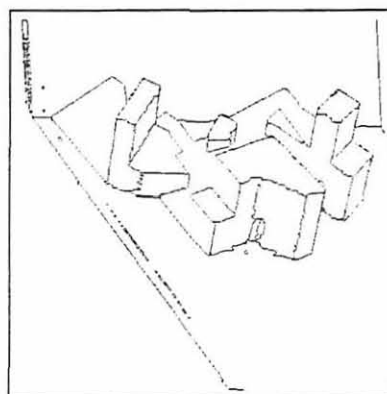
(c) Segmentação manual



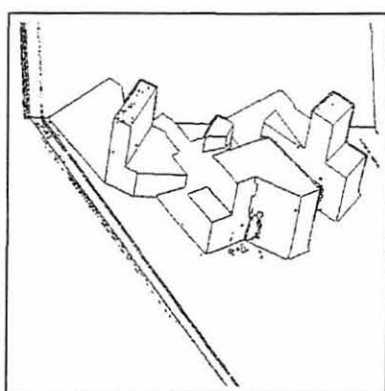
(d) Segmentação por detecção de bordas



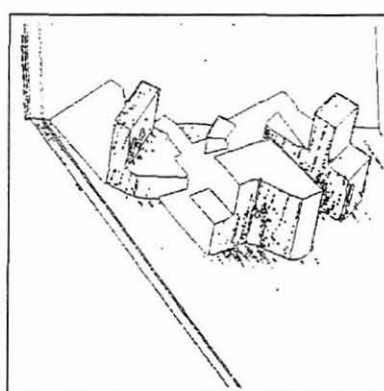
(e) WSU



(f) UB



(g) UE

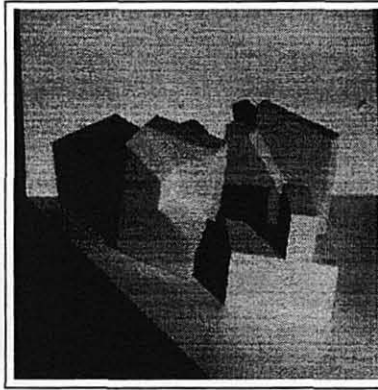


(h) USF

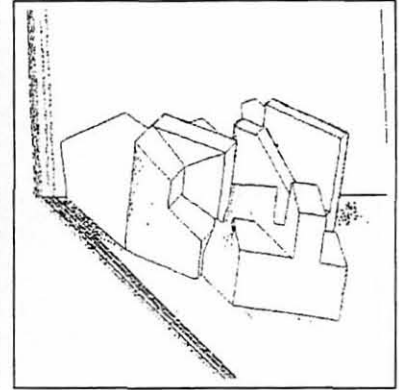
Figura 5.4: Imagem de treinamento ABW (*abw.train.1*).



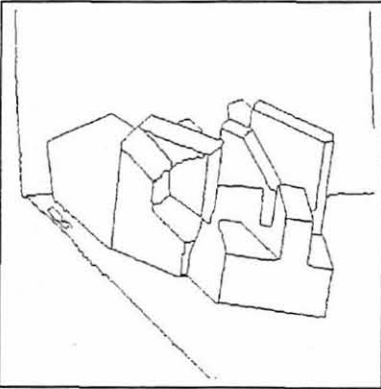
(a) Imagem de profundidade



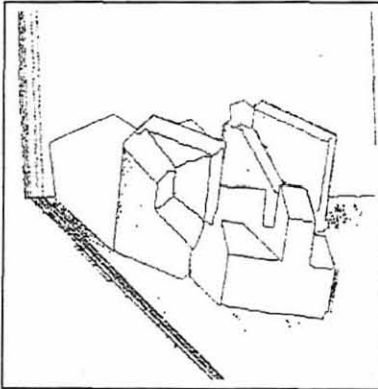
(b) Imagem de intensidade



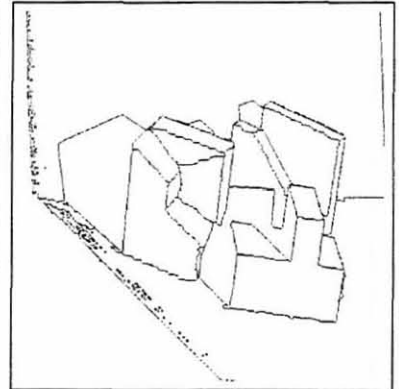
(c) Segmentação manual



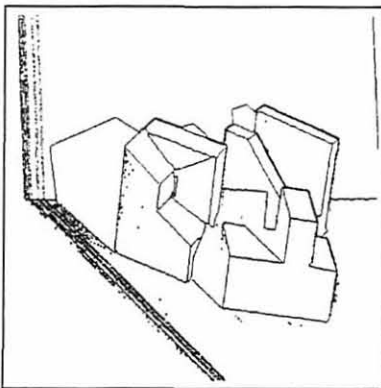
(d) Segmentação por detecção de bordas



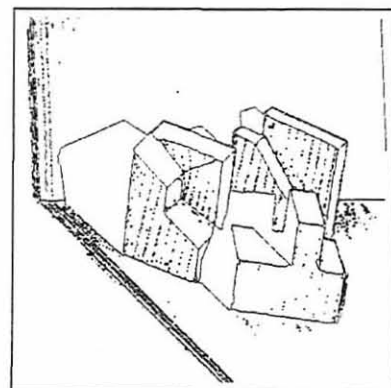
(e) WSU



(f) UB



(g) UE



(h) USF

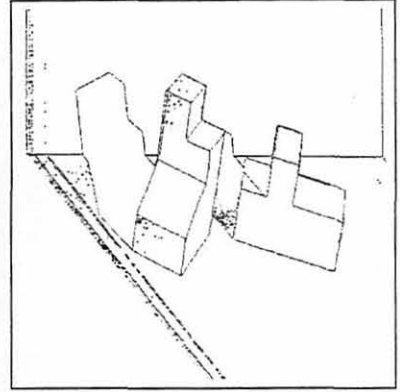
Figura 5.5: Imagem de treinamento ABW (*abw.train.2*).



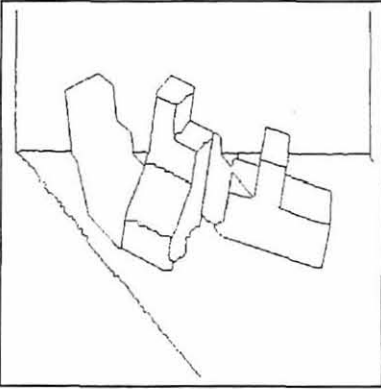
(a) Imagem de profundidade



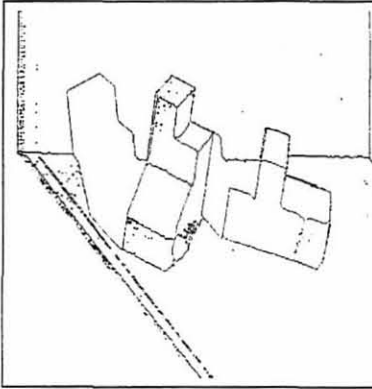
(b) Imagem de intensidade



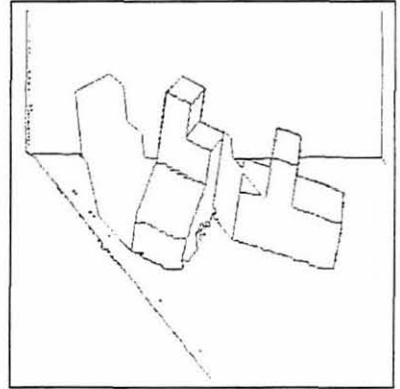
(c) Segmentação manual



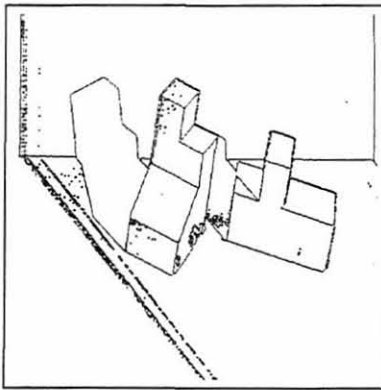
(d) Segmentação por detecção de bordas



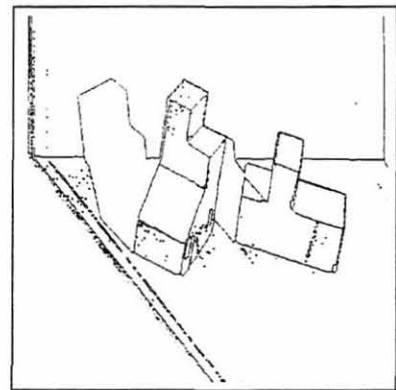
(e) WSU



(f) UB



(g) UE

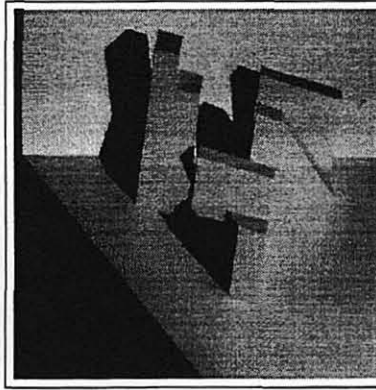


(h) USF

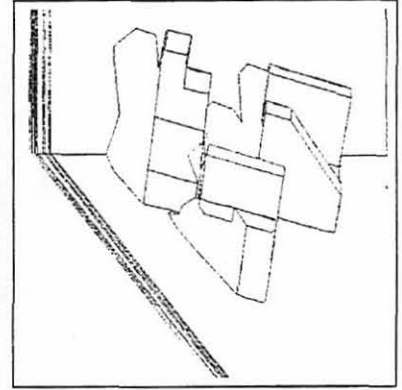
Figura 5.6: Imagem de treinamento ABW (*abw.train.3*).



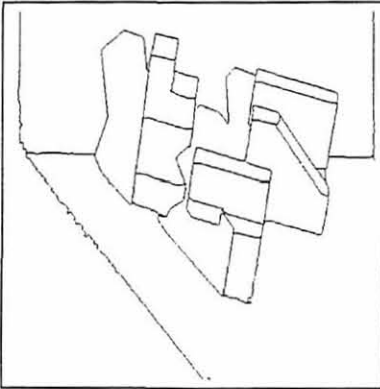
(a) Imagem de profundidade



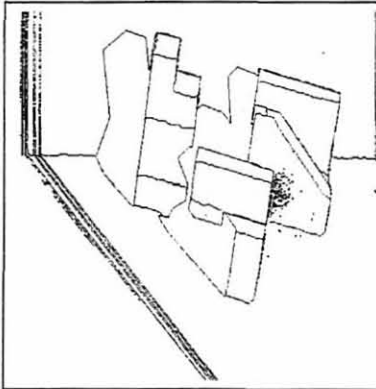
(b) Imagem de intensidade



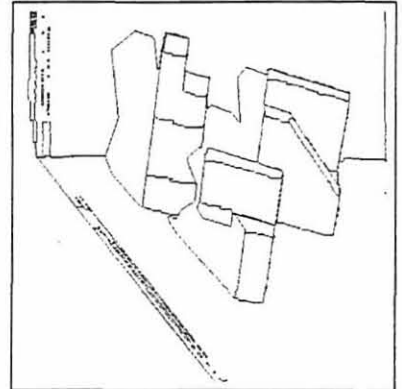
(c) Segmentação manual



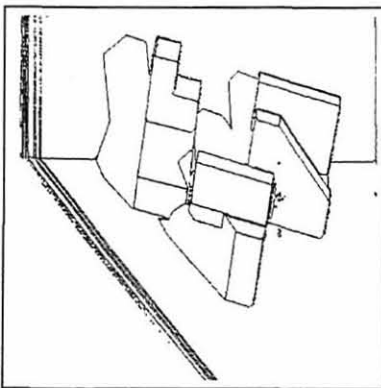
(d) Segmentação por detecção de bordas



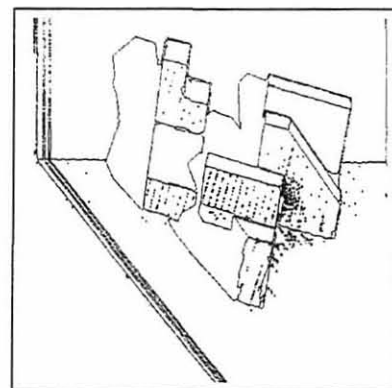
(e) WSU



(f) UB



(g) UE

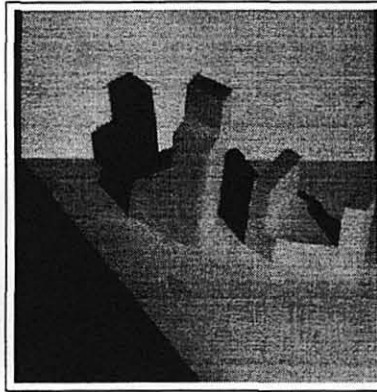


(h) USF

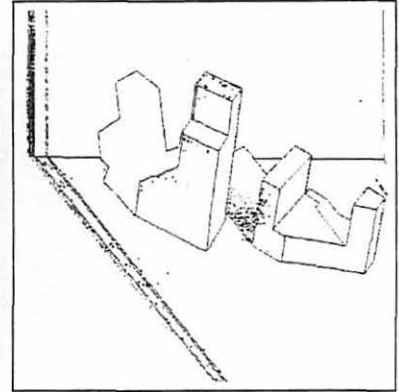
Figura 5.7: Imagem de treinamento ABW (*abw.train.4*).



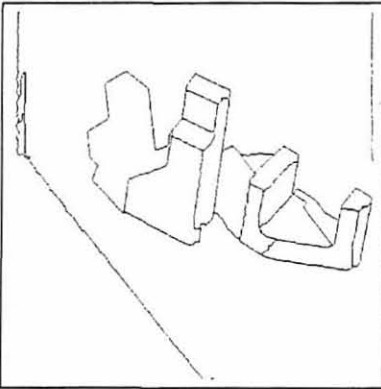
(a) Imagem de profundidade



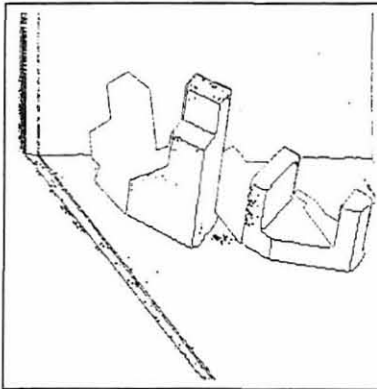
(b) Imagem de intensidade



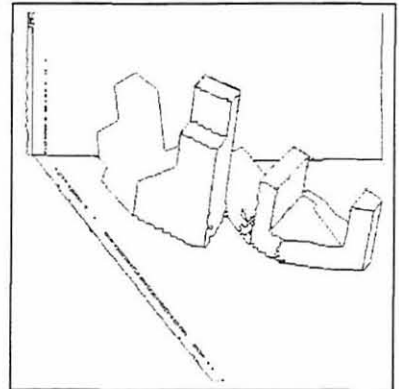
(c) Segmentação manual



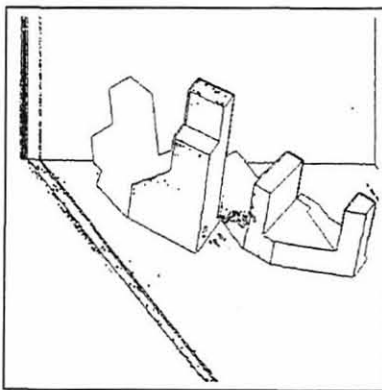
(d) Segmentação por detecção de bordas



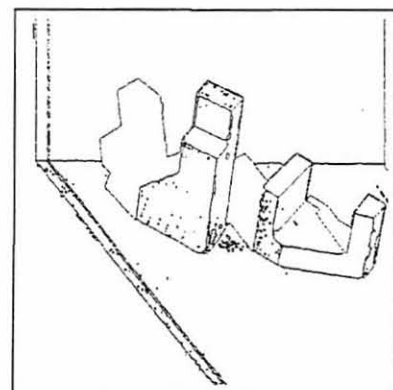
(e) WSU



(f) UB



(g) UE

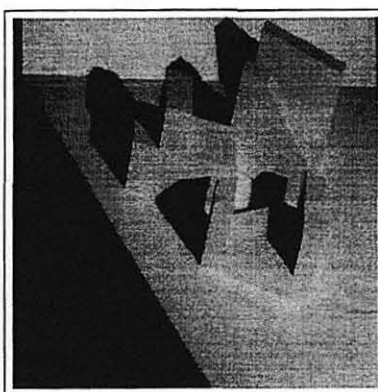


(h) USF

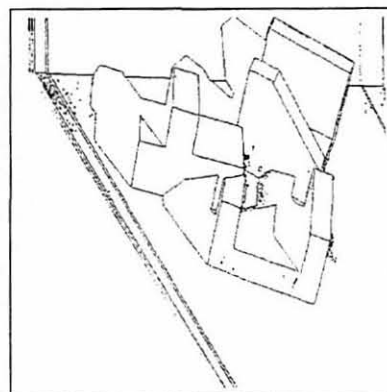
Figura 5.8: Imagem de treinamento ABW (*abw.train.5*).



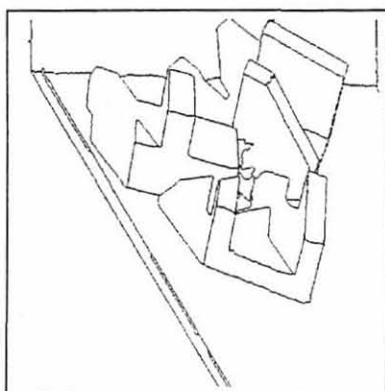
(a) Imagem de profundidade



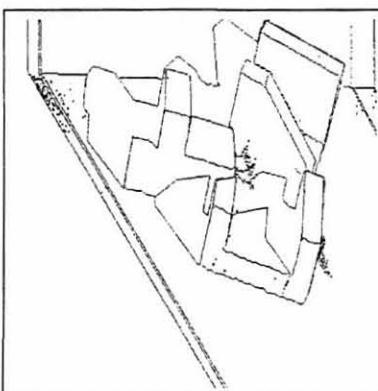
(b) Imagem de intensidade



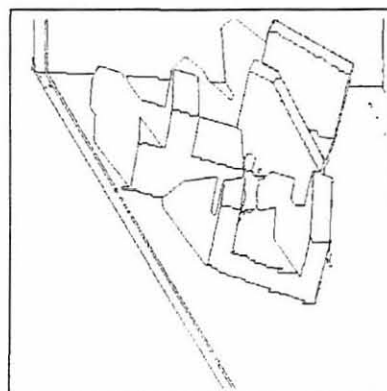
(c) Segmentação manual



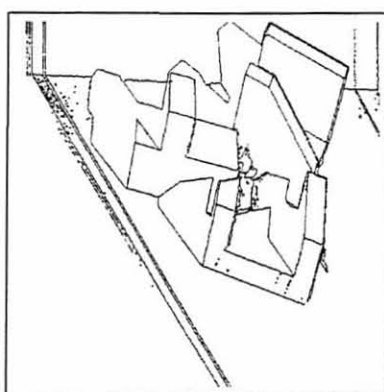
(d) Segmentação por detecção de bordas



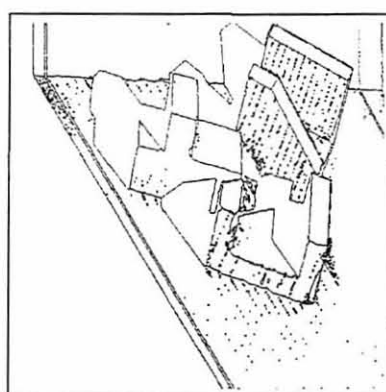
(e) WSU



(f) UB



(g) UE

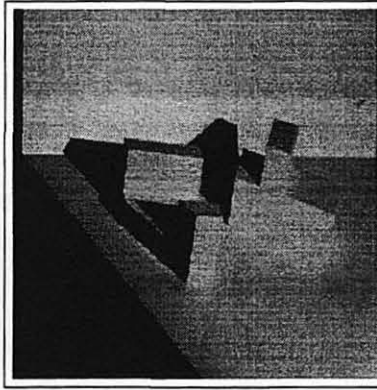


(h) USF

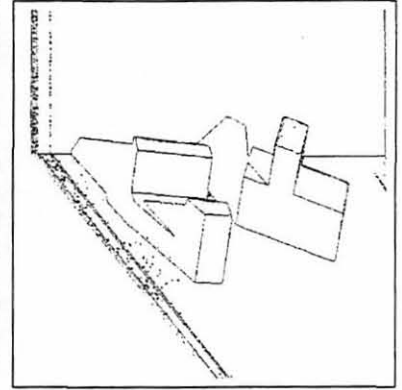
Figura 5.9: Imagem de treinamento ABW (*abw.test.1*).



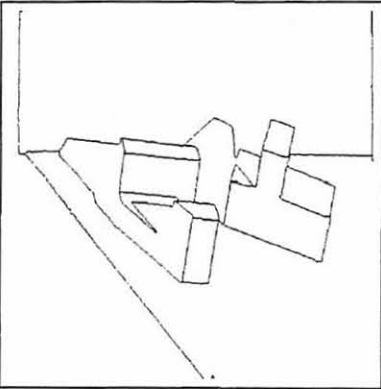
(a) Imagem de profundidade



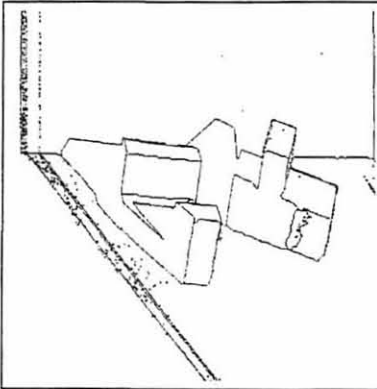
(b) Imagem de intensidade



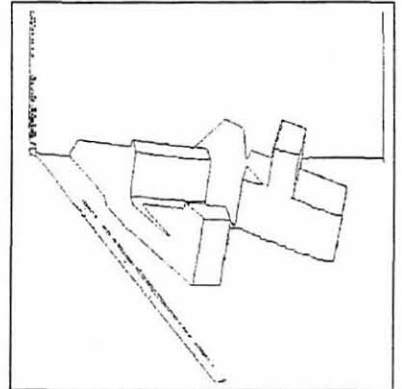
(c) Segmentação manual



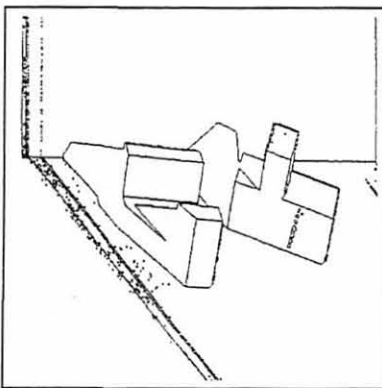
(d) Segmentação por detecção de bordas



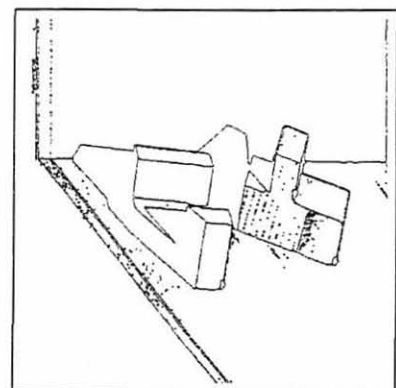
(e) WSU



(f) UB

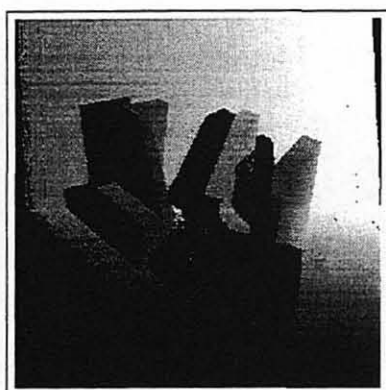


(g) UE

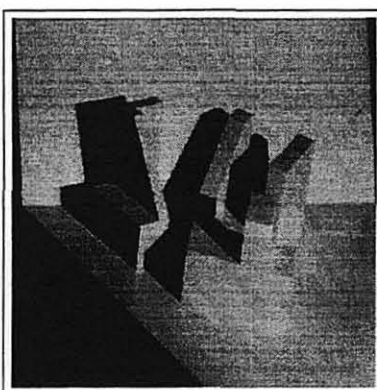


(h) USF

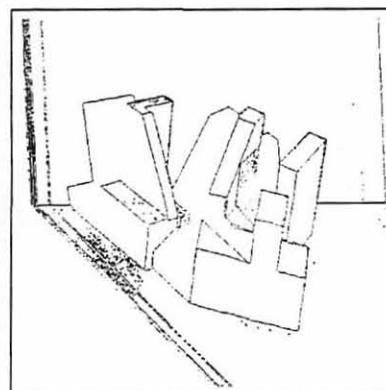
Figura 5.10: Imagem de treinamento ABW (*abw.test.3*).



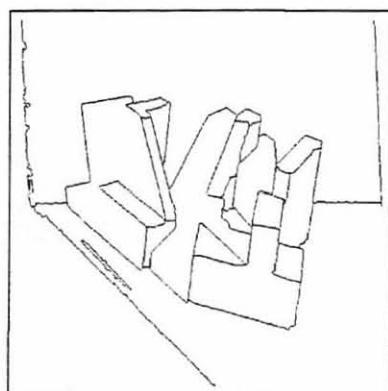
(a) Imagem de profundidade



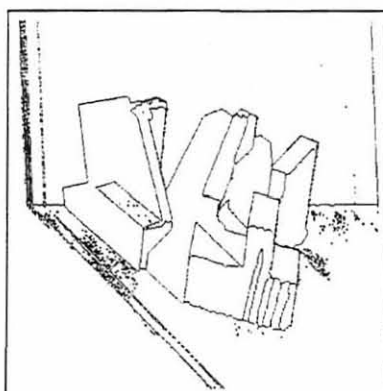
(b) Imagem de intensidade



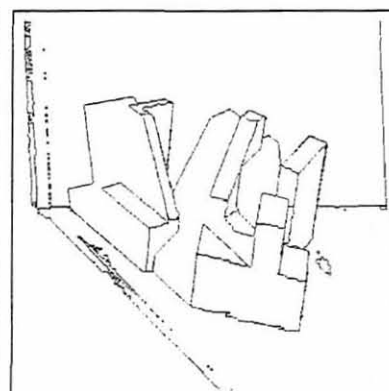
(c) Segmentação manual



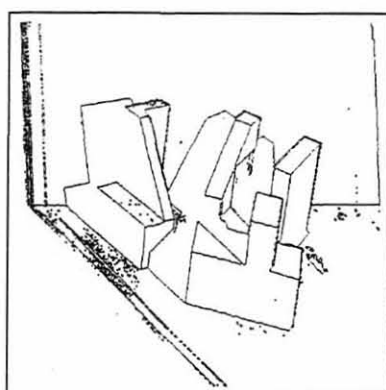
(d) Segmentação por detecção de bordas



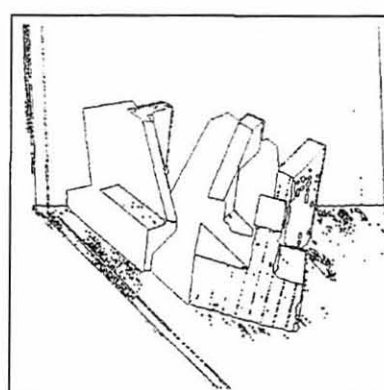
(e) WSU



(f) UB



(g) UE

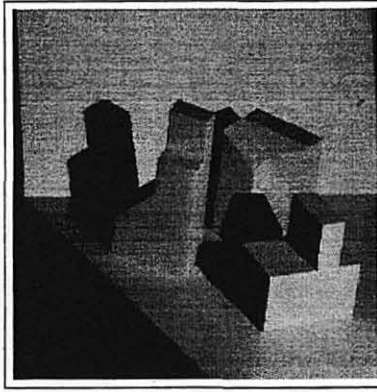


(h) USF

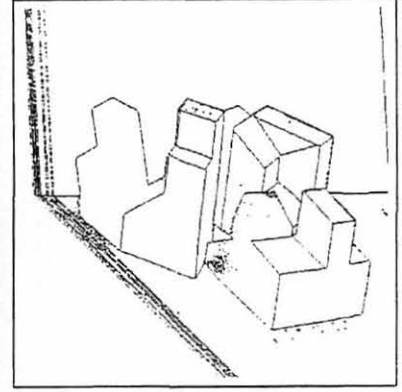
Figura 5.11: Imagem de treinamento ABW (*abw.test.4*).



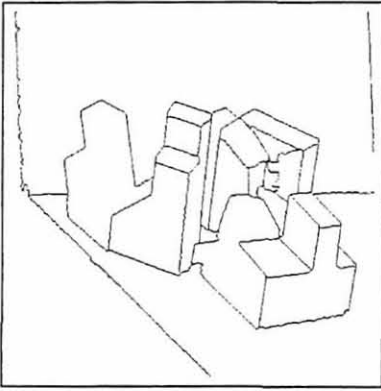
(a) Imagem de profundidade



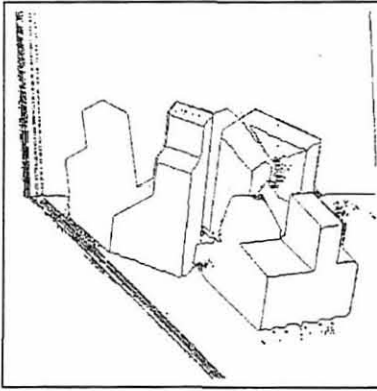
(b) Imagem de intensidade



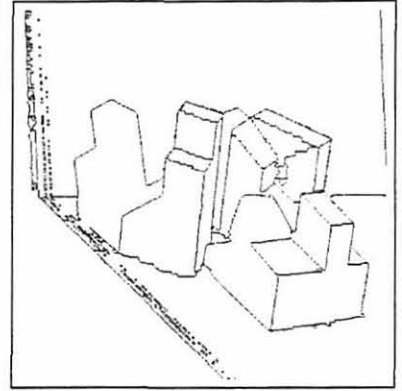
(c) Segmentação manual



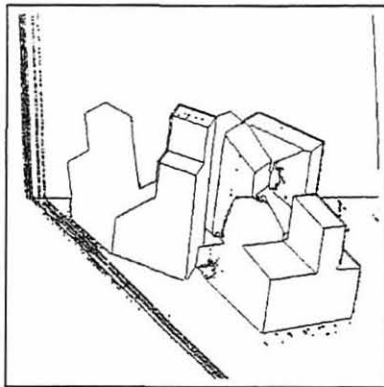
(d) Segmentação por detecção de bordas



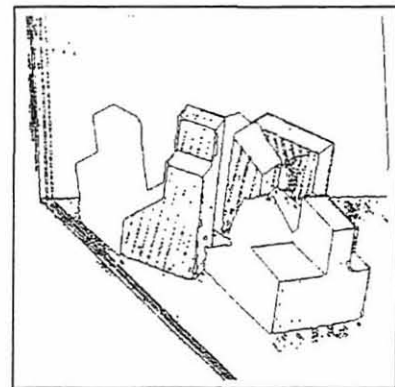
(e) WSU



(f) UB



(g) UE

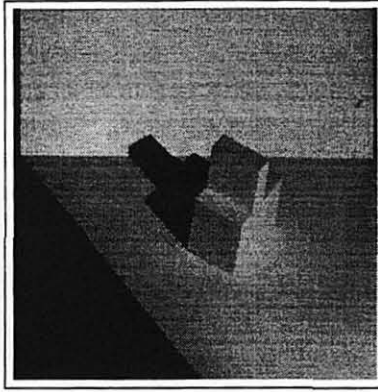


(h) USF

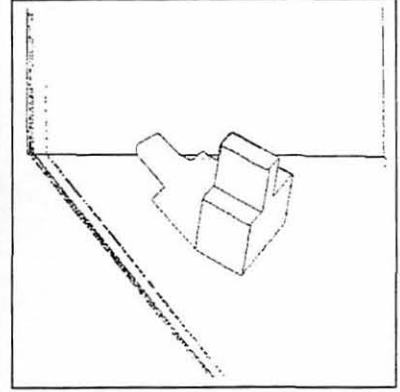
Figura 5.12: Imagem de treinamento ABW (*abw.test.8*).



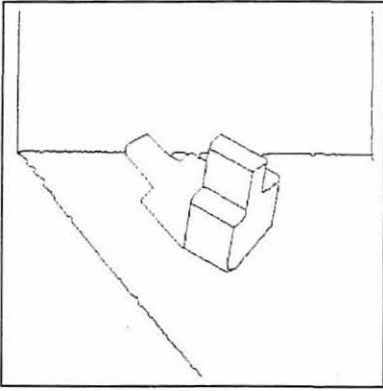
(a) Imagem de profundidade



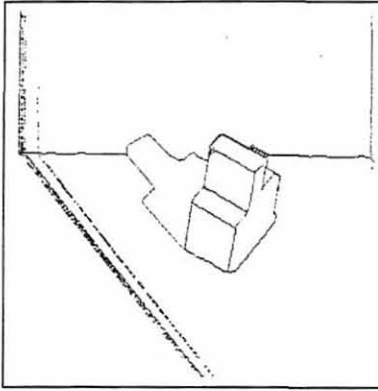
(b) Imagem de intensidade



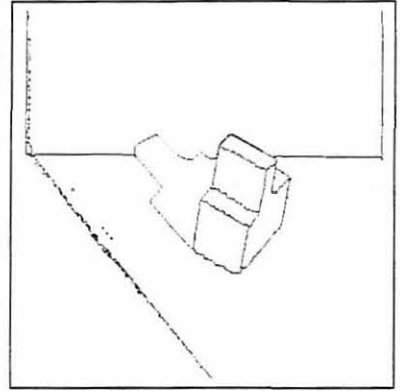
(c) Segmentação manual



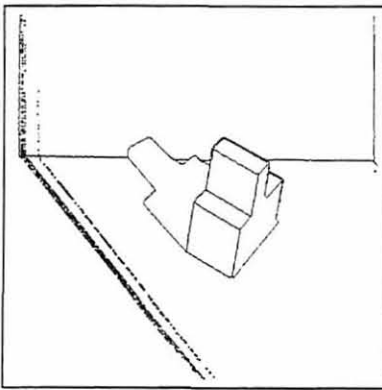
(d) Segmentação por detecção de bordas



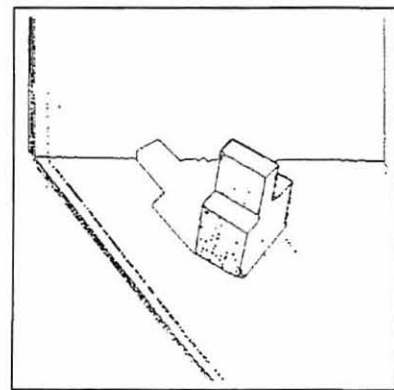
(e) WSU



(f) UB



(g) UE

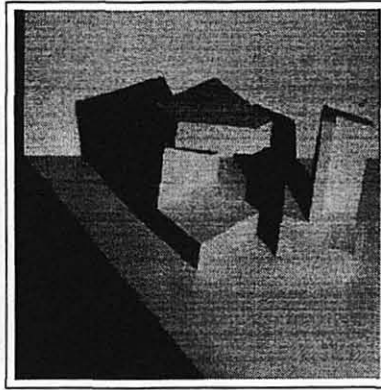


(h) USF

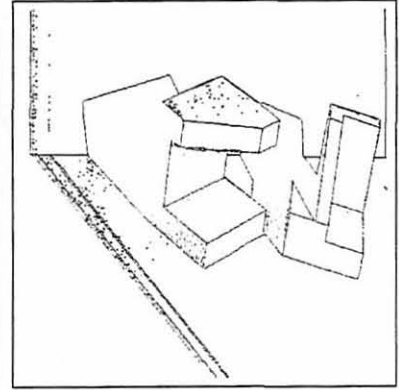
Figura 5.13: Imagem de treinamento ABW (*abw.test.10*).



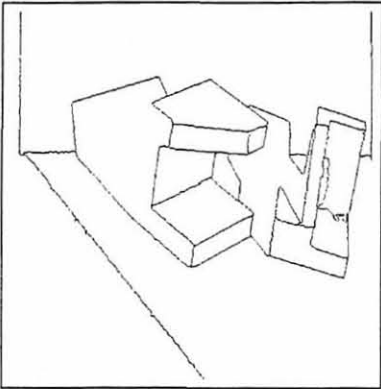
(a) Imagem de profundidade



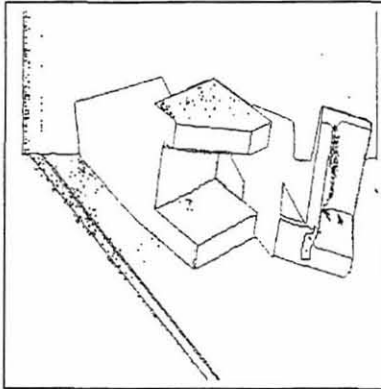
(b) Imagem de intensidade



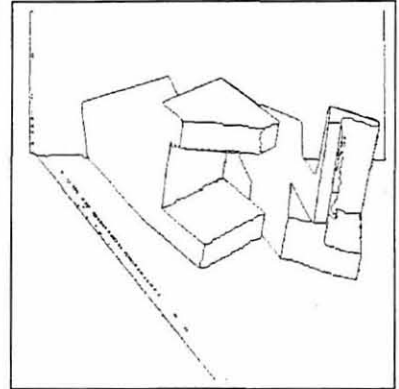
(c) Segmentação manual



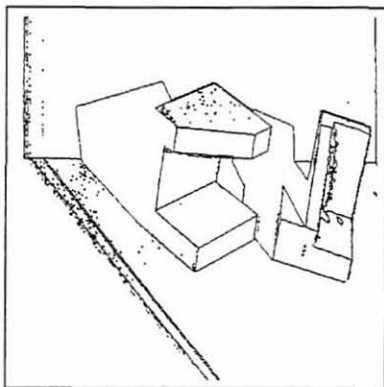
(d) Segmentação por detecção de bordas



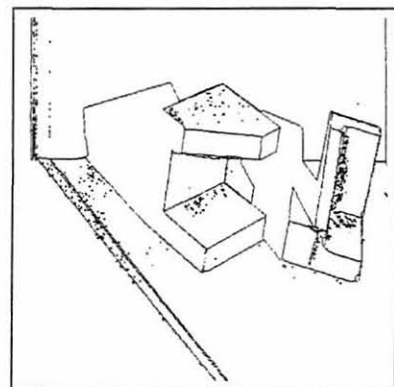
(e) WSU



(f) UB



(g) UE



(h) USF

Figura 5.14: Imagem de treinamento ABW (*abw.test.18*).

5.4 Conclusão

Os resultados experimentais mostrados apontam os pontos positivos e negativos de cada um dos algoritmos discutidos. Nenhum deles apresentou um resultado definitivo para o problema da segmentação de imagens de profundidade.

O algoritmo UB apresenta uma perda grande com relação à localização precisa das bordas entre as regiões da imagem, dificultando um processo de representação dessa imagem, porém é um método extremamente rápido e que dificilmente será superado neste sentido. Já o algoritmo UE apresentou bons resultados com relação à descrição topológica das superfícies, porém é sensível a ruídos, principalmente nas regiões de bordas de profundidade. O algoritmo USF é extremamente sensível a ruídos e não fornece uma segmentação eficiente com regiões bem delimitadas.

Com relação ao tempo de processamento, o algoritmo WSU foi extremamente lento e não apresentou bons resultados com relação a preservação da topologia e sensibilidade a ruídos.

Não foi possível fazer uma comparação aprofundada com os tempos de processamento de cada algoritmo, porque alguns foram desenvolvidos em plataformas diferentes com códigos muito dependentes, dificultando assim a portabilidade e avaliações de desempenho. No trabalho de Hoover são mostradas os tempos de processamento de cada algoritmo para plataformas específicas onde os algoritmos foram desenvolvidos e testados, porém não é feita nenhuma comparação conclusiva com relação ao desempenho.

O método de segmentação desenvolvido baseado em detecção de bordas, obteve resultados satisfatórios com relação aos objetivos pretendidos neste trabalho e obteve um tempo de processamento abaixo da média de todos os algoritmos com exceção do algoritmo UB.

CAPÍTULO 6

CONCLUSÃO

Neste capítulo são apresentadas as conclusões deste trabalho, assim como suas contribuições e as sugestões para trabalhos futuros.

A extração de características e segmentação de imagens são pontos críticos para a análise de imagens em sistemas de visão computacional. Dessa forma, é necessário o profundo conhecimento das técnicas já empregadas e o desenvolvimento de novas soluções, especialmente em imagens de profundidade. Essas imagens, que fornecem a informação 3D de forma explícita, abriram um novo conjunto de desafios para os pesquisadores das mais diversas áreas.

Dentro deste contexto, o trabalho apresentou um novo método para segmentação de imagens de profundidade por técnicas de detecção de bordas. O método foi devidamente comparado com os quatro principais segmentadores encontrados na literatura, incluindo diversos resultados experimentais.

Os principais pontos positivos do trabalho desenvolvido podem ser observados nos resultados apresentados, que mostraram uma segmentação que preserva a topologia das superfícies dos objetos. Nestes resultados, a eliminação dos ruídos nas imagens segmentadas também pode ser considerado como um ponto positivo. Além disso, a eliminação de muitos limiares fixos ajuda na generalização do algoritmo para ser aplicado a diversos tipos de imagens de entrada. Finalmente, ressalta-se que o método não é restrito a aplicação em imagens compostas apenas por objetos poliédricos.

As principais contribuições deste trabalho são:

1. Novo método para segmentação de imagens de profundidade baseado em detecção de bordas.
2. Algoritmo aplicado também a objetos não poliédricos.
3. Estrutura modular do algoritmo facilita o paralelismo.

4. O desenvolvimento do cálculo de limiar automático para a detecção de bordas.
5. A abordagem desenvolvida para a classificação dos *pixels* válidos no processo de cálculo dos vetores normais.
6. A criação de um algoritmo híbrido de afinamento baseado em métodos de esqueletização por transformação do eixo médio e em elementos estruturantes de Morfologia Matemática.
7. A elaboração de um processo de fechamento do mapa de bordas a partir dos *pixels* de extremidade em um *pixel* de borda próximo, levando em consideração o *pixel* de borda mais apropriado, baseado na informação de profundidade.
8. Segmentação com regiões significativas eliminando as regiões de ruído e preservando a topologia.

A partir deste trabalho, duas linhas de atuação podem ser identificadas, sendo aqui propostas como trabalhos futuros. A primeira delas diz respeito à detecção das bordas de orientação que, como pode ser observado nos resultados experimentais, ainda apresenta uma série de deficiências. Neste caso, para se obter resultados com mais qualidade, é necessário realizar alguns melhoramentos no processo. Para isso, um caminho é atuar de forma a eliminar ainda mais o efeito dos ruídos encontrados nas bordas de orientação, e assim reduzir o tempo de processamento com afinamentos e fechamentos no mapa de bordas. Desta forma, pode-se obter um segmentador mais eficiente e robusto.

Uma outra linha muito promissora é a construção de representações para os objetos segmentados. Para isso, existem três grandes desafios:

1. Construção de representação para os objetos segmentados.
2. Ajuste de superfícies às regiões definidas.
3. Combinação das superfícies que compõem cada objeto observado na imagem.
4. Adequação de modelos para os objetos.

5. Segmentação de várias vistas e integração dos resultados numa representação 3D completa.

Os resultados dessas linhas de atuação podem levar ao melhoramento dos processos de interpretação e reconhecimento de objetos e, portanto, contribuir decisivamente para o avanço da área de visão computacional.

BIBLIOGRAFIA

- [1] A.P. Ashbrook, R.B. Fisher, Robertson. C., e N. Werghi. Segmentation of range data into rigid subsets using planar surface patches. *Proc. British Machine Vision Conference BMVC97*, páginas 530–539, 1997.
- [2] A.P. Ashbrook, R.B. Fisher, Robertson. C., e N. Werghi. Segmentation of range data into rigid subsets using surface patches. *Proceedings of Int. Conf. on Computer Vision*, páginas 201–206, 1998.
- [3] O.R.P. Bellon. *Imagens de profundidade: segmentação e descrição por superfícies planares*. Tese de Doutorado, DCA-FEEC, UNICAMP, 1997.
- [4] O.R.P. Bellon, A.I. Direne, e L. Silva. Building meaningful edge maps from range images. *ICIAP'99, Proceedings of 10th International Conference on Image Analysis and Processing, Venice, Italy*, páginas 227–232. International Association for Pattern Recognition, IEEE, setembro de 1999.
- [5] O.R.P. Bellon, A.I. Direne, e L. Silva. Edge detection to guide range image segmentation by clustering techniques. *ICIP'99, Proceedings of 6th IEEE International Conference on Image Processing, Kobe, Japan*. IEEE, novembro de 1999. 26PP1.9.
- [6] P.J. Besl. *Advances in Machine Vision*, capítulo 1. Active Optical Range Imaging Sensors, páginas 1–63. Springer-Verlag, 1989.
- [7] P.J. Besl e R.C. Jain. Segmentation through variable-order surface fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(2):167–192, fevereiro de 1988.
- [8] S.M. Bhandarkar e A. Siebert. Integrating edge and surface information for range image segmentation. *Pattern Recognition*, 25:947–962, 1992.

- [9] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, novembro de 1986.
- [10] T.M. Celinski. Métodos de agrupamento: Uma abordagem comparativa com aplicação em segmentação de imagens de profundidade. Dissertação de Mestrado, DInf-PPGInf, UFPR, 1998.
- [11] R. Deriche. Using Canny’s criteria to derive a recursively implemented optimal edge detector. *IJCV*, 1(2):167–187, 1987.
- [12] R. Dubes e A.K. Jain. Clustering techniques: the user’s dilemma. *Pattern Recognition*, 8:247–260, 1976.
- [13] R.O. Duda e P.E. Hart. *Pattern Classification and Scene Analysis*. Wiley Interscience, 1973.
- [14] P.J. Flynn e Jain A.K. Bonsai: 3D object recognition using constrained search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1066–1075, outubro de 1991.
- [15] P.J. Flynn e A.K. Jain. Surface classification: Hypothesis testing and parameter estimation. *CVPR*, páginas 261–267, junho de 1988.
- [16] J.D. Foley. *Computer Graphics*. Addison-Wesley, 1996.
- [17] R.C. Gonzalez e R.E. Woods. *Digital Image Processing*. Addison-Wesley, 1993.
- [18] J.F. Haddon. Generalized threshold selection for edge detection. *Pattern Recognition*, 21:195–203, 1988.
- [19] M.D. Heath. Robust visual methods for accessing the relative performance of edge detection algorithms. Dissertação de Mestrado, Departament of Computer Science and Engineering, University of South Florida, USA, 1996.

- [20] R. Hoffman e A.K. Jain. Segmentation and classification of range images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):608–620, setembro de 1987.
- [21] A. Hoover, G. Jean-Baptiste, X.Y. Jiang, P.J. Flynn, H. Bunke, D.B. Goldgof, K. Bowyer, D.W. Eggert, A. Fitzgibbon, e R.B. Fisher. An experimental comparison of range image segmentation algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7):673–689, julho de 1996.
- [22] A.K. Jain e P.J. Flynn. Image segmentation using clustering. *AIU96*, páginas 65–83, 1996.
- [23] R.A. Jarvis. A perspective on range finding techniques for computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):122–139, 1983.
- [24] X. Jiang e H. Bunke. Edge detection in range images based on scan line approximation. *Computer Vision and Image Understanding*, 2(73):183–199, fevereiro de 1999.
- [25] X. Jiang e H. Nunke. Fast segmentation of range images into planar regions by scan line grouping. *Machine Vision and Applications*, 7(2):115–122, 1994.
- [26] R. Krishnapuram e S. Gupta. Morphological methods for detection and classification for edges in range images. *Journal of Mathematical Imaging and Vision*, 2:351–375, 1992.
- [27] L. Lam, S. Lee, e C.Y. Suen. Thinning methodologies - a comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(9):869–885, 1992.
- [28] G. Malandain e S. Fernandez-Vidal. Euclidean skeletons. *IMAGE AND VISION COMPUTING*, 16(5):317–327, abril de 1998.
- [29] D. Marr e E.C. Hildreth. Theory of edge detection. *Proceedings of Royal Society of London*, B-207:187–217, 1980.

- [30] D. Mintz. Robust consensus based edge-detection. *CVGIP - Image Understanding*, 59(2):137–153, março de 1994.
- [31] A. Mitiche e J.K. Aggarwal. Detection of edges using range information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):174–178, março de 1983.
- [32] V.S. Nalwa e T.O. Binford. On detecting edges. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):699–714, novembro de 1986.
- [33] W.K. Pratt. *Digital Image Processing*. Wiley Interscience, 1991.
- [34] W.H. Press, S.A. Teukolsky, Vetterling W.T., e B.P. Flannery. *Numerical Recipes in C - The Art of Scientific Computing*. Cambridge University Press, second edition, 1992.
- [35] C. Rothwell, J. Mundy, B. Hoffman, e V. Nguyen. Driving vision by topology. Relatório Técnico 2444, INRIA, dezembro de 1994.
- [36] J.C. Russ. *The image processing handbook*. CRC Press, second edition, 1995.
- [37] P.K. Sahoo, S. Soltani, e A.K.C. Wong. A survey of thresholding techniques. *Computer Vision Graphics and Image Processing*, (41):233–260, 1988.
- [38] J. Serra. *Image Analysis and Mathemaical Morphology*. Academic Press, 1982.
- [39] L. Silva, D. Weingaertner, e O.R.P. Bellon. Ferramentas para detecção de bordas em imagens de profundidade. *Anais do XVIII CSBC - SEMISH/CTD/CTIC/WEI/WIE, Belo Horizonte, Brasil*, volume 1, páginas 417–429. SBC, agosto de 1998.
- [40] T.G Stahs e F.M. Wahl. Fast and robust range data acquisition in a low-cost environment. *SPIE #1395: Close-Range Photogrammetry Meets Machine Vision*, páginas 496–503, 1990. Zurich.

- [41] Y.F. Tsao e K.S. Fu. Parallel thinning operations for digital binary images. *Proceedings CVPR*, páginas 150–155, 1981.